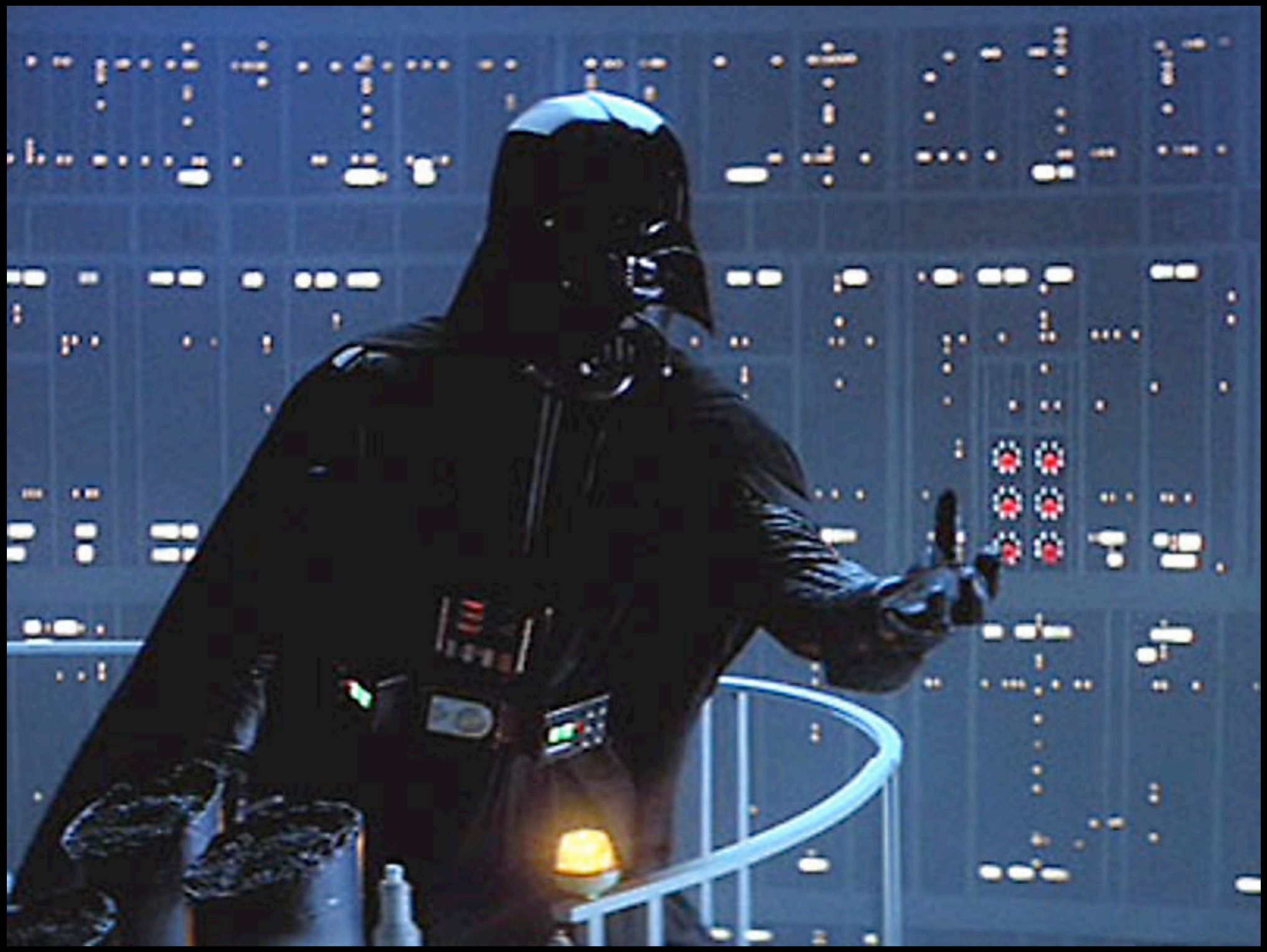
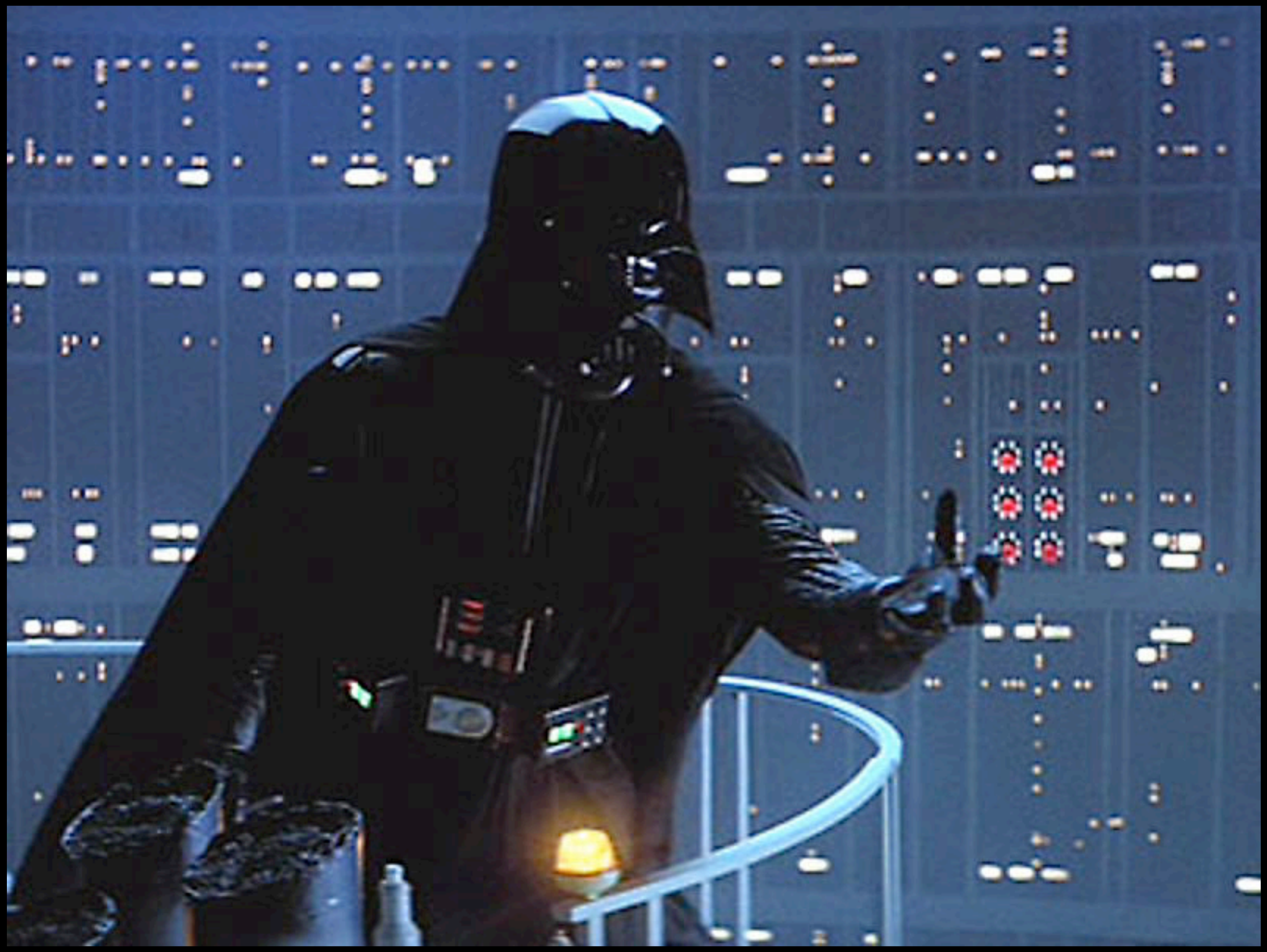
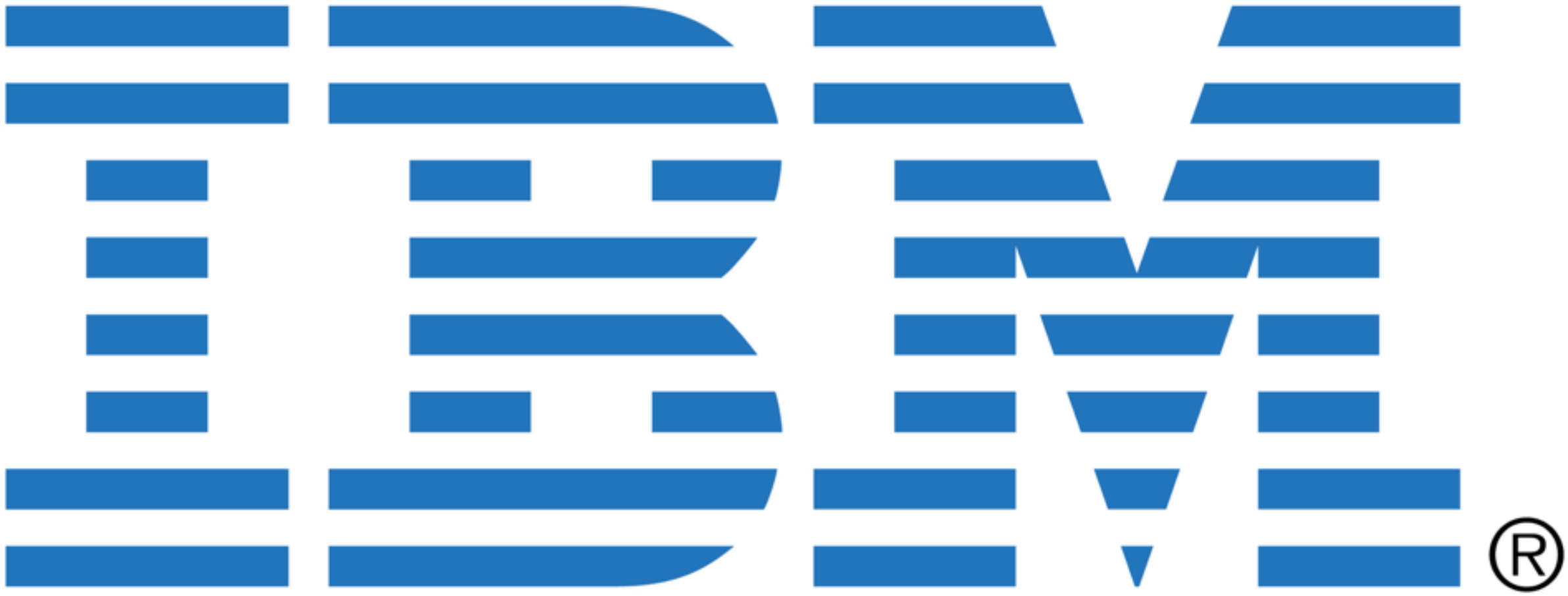
Design Patterns for Mobile Apps

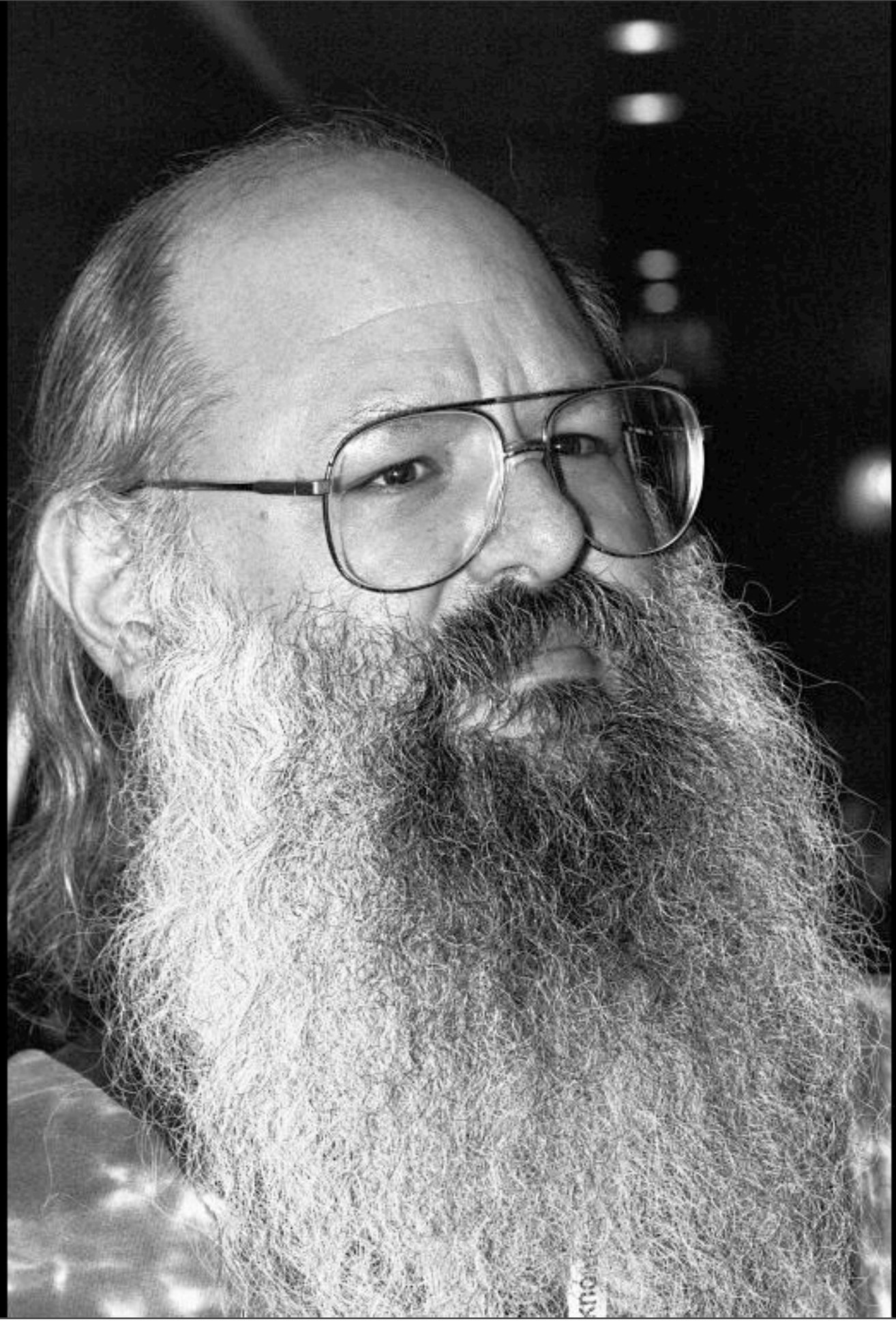
Cocoa Design Patterns

NSBrief





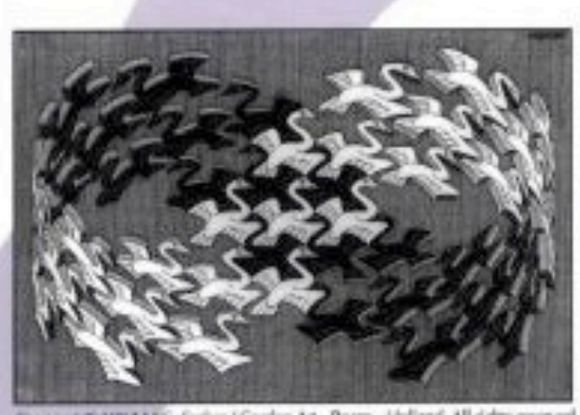




Design Patterns

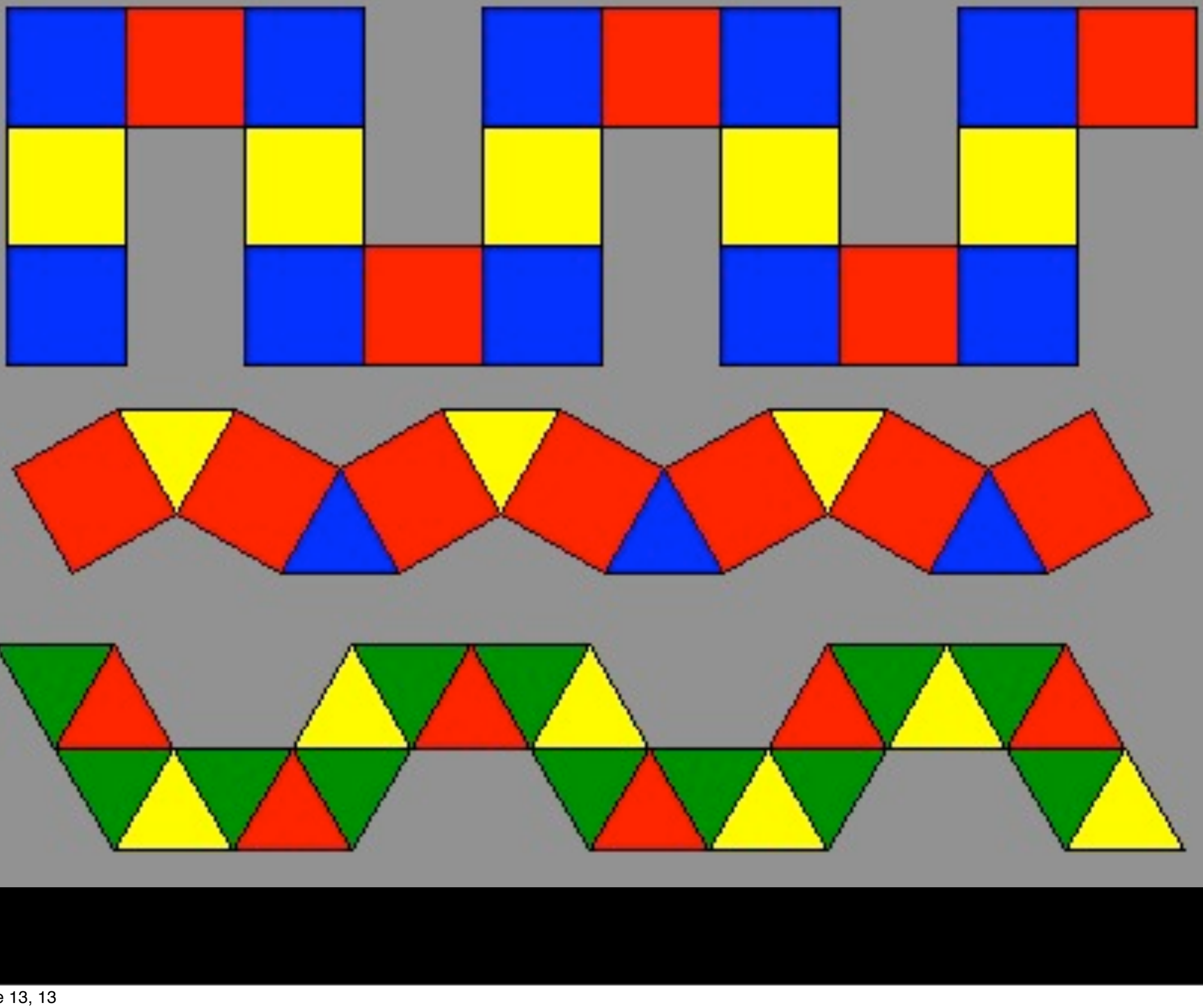
Elements of Reusable Object-Oriented Software

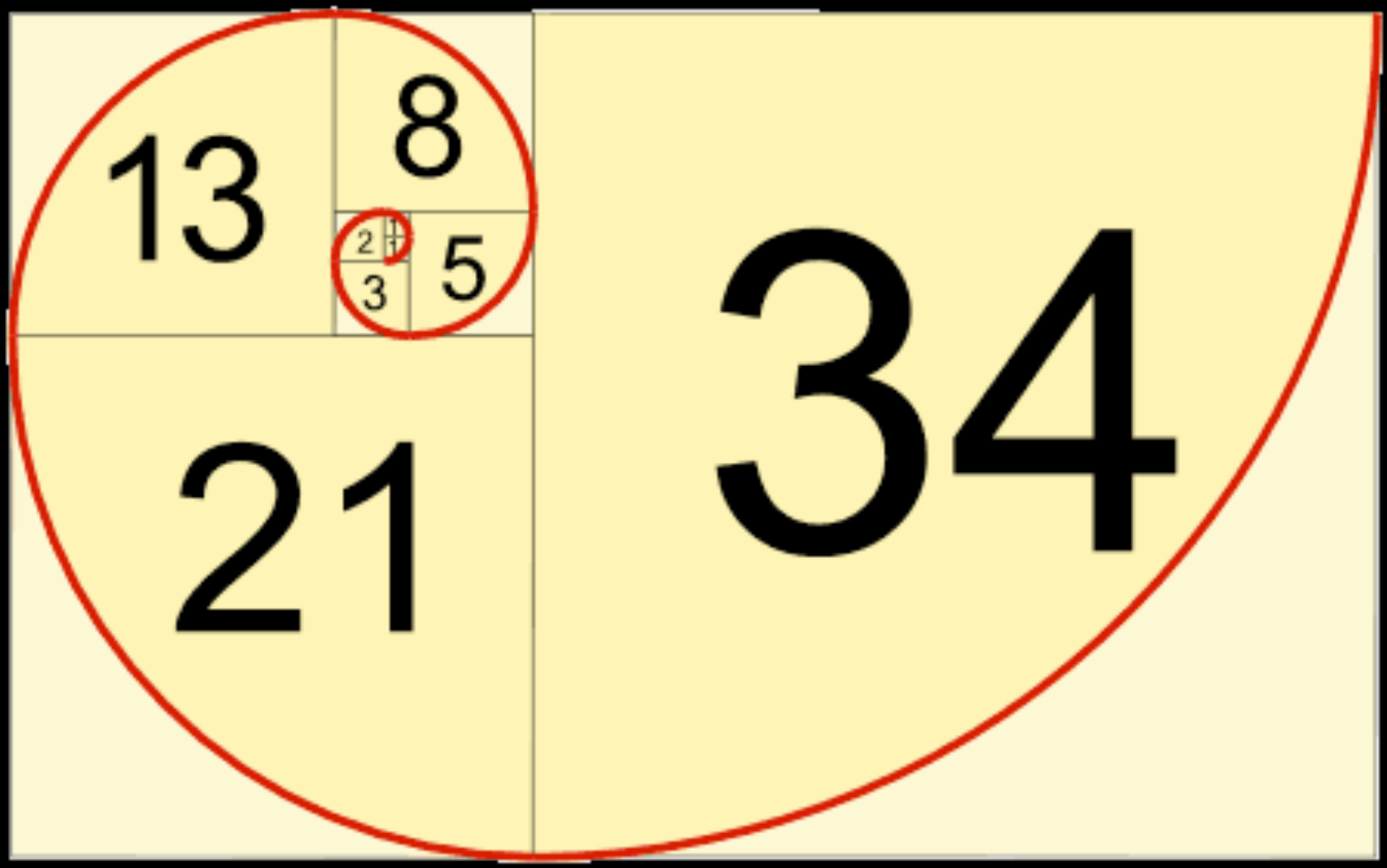
Erich Gamma
Richard Helm
Ralph Johnson
John Vlissides



Foreword by Grady Booch

ADDISON-WESLEY PROFESSIONAL COMPUTING SERIES




13
8
2
1
3
5
34
21

20/200	O	1
20/100	M G	2
20/70	W T F	3
20/50	S T F U	4
20/40	P W N 3 D	5
20/30	U R A N 0 0 B	6
20/20	L M A O R O T F	7
20/15	K T H X B Y E :P	8

IF YOU CAN READ THIS, UR EYEZ R TEH 1337. TTYL.


© 2001 Universal Pictures

Hawtness

853 apps per day

148apps.biz
January 2013

133 games per day

148apps.biz
January 2013

Quality Takes Time

Time == $$$

Time Tested Solutions

Quiche £3.50
..... £4.00
..... £4.30
..... £4.30
..... £4.30

farmhouse lunches

All served with French Bread, Pickles and Garnished with Salad

Your choice of our own cooked meats:

Ham £6.50
Honey Roast Ham £6.50
Beef £6.50
Pork £6.50
Turkey £6.50
Quarter Chicken £6.50
2 Sausage Rolls £5.50
1 Pastie £5.50

or:

Cheddar £5.30
Stilton £5.30
Brie £5.30

hot roast lunches

These dishes include New Potatoes, Cauliflower cheese, Carrots and Peas

Roast Pork with Crackling £8.50
Rare Roast Beef £8.50
Roast Lamb £8.50
Roast Turkey Breast £8.50

casseroles

All served with Jacket or New Potatoes

Venison and Boar £9.00
Vicars £9.00
Sweeney's £9.00

Sweeney & Todd

hot pies

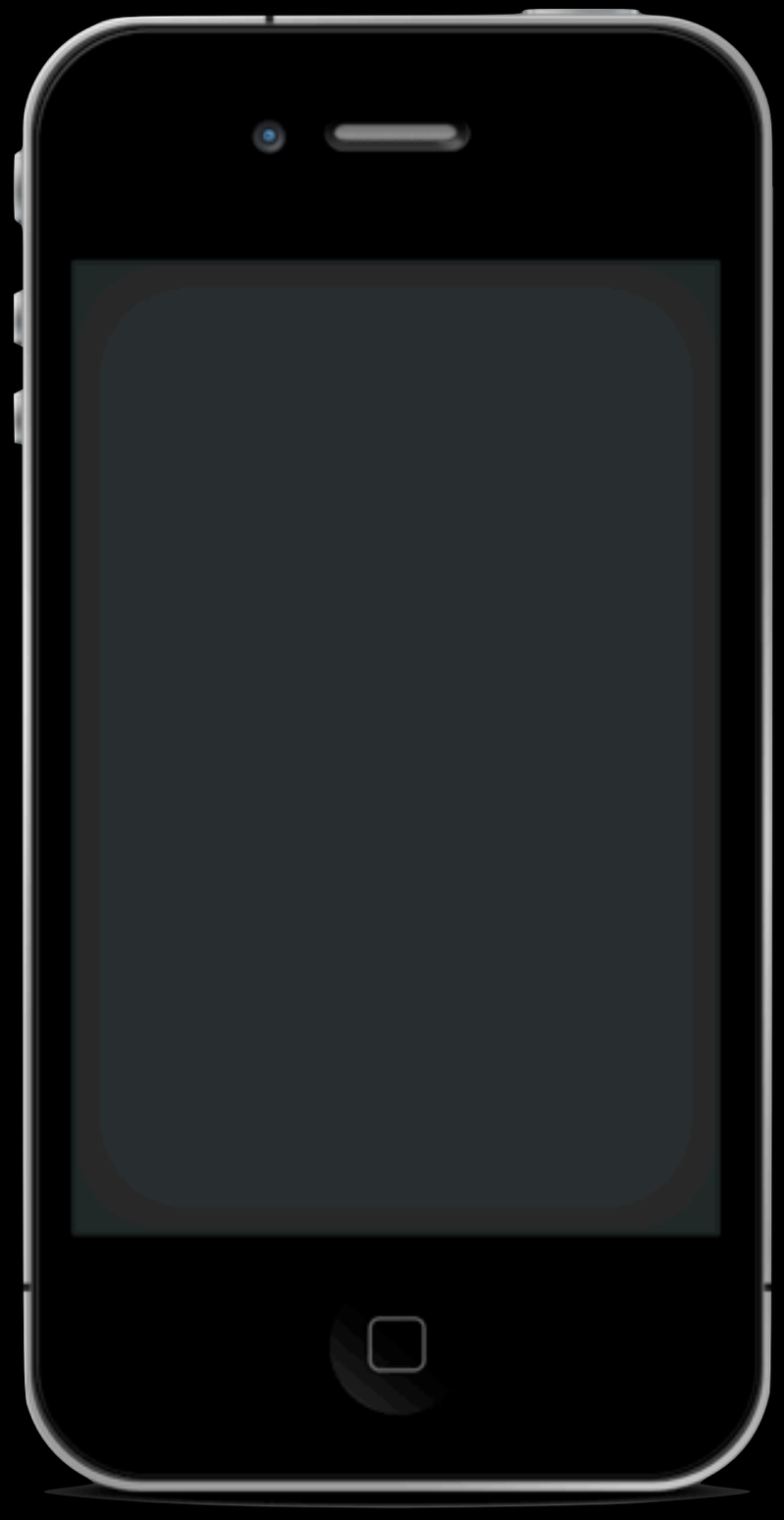
Please ask the Waitress for the Daily Selection from:

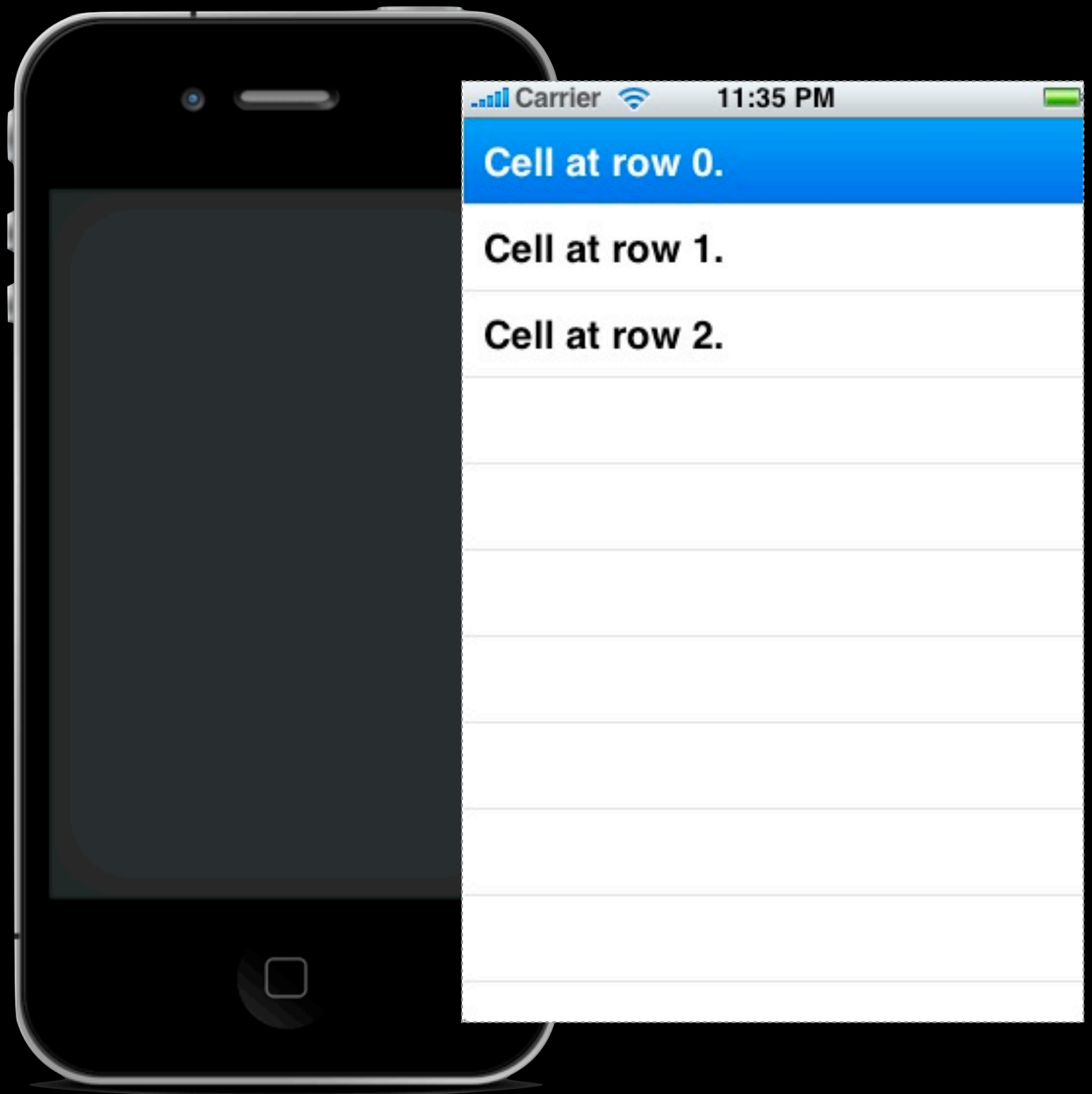
Steak £6.50
Steak and Oyster £6.50
Steak and Mushroom £6.50
Steak and Kidney £6.50
Minced Beef and Onion £6.50
Yorkshire Ham and Stilton £6.50
Chicken and Sweetcorn £6.50
Lamb and Mint £6.50
Pork and Apple £6.50
Chicken, Honey and Mustard £6.50
Venison and Boar £6.50
Rump Steak and Stilton £6.50
Chicken, Broccoli and Stilton £6.50
Beef and Horseradish £6.50
Chicken and Leek £6.50
Cheese and Vegetable £6.50
Five Nations £6.50
Vicars £6.50
Steak and Ale £6.50
Chicken Cajun £6.50
Sweeney's £6.50
Turkey and Bacon £6.50
Chicken and Ham £6.50

PLUS VARIOUS OTHERS


COLA
12 FL. OZ. 355 mL
Cola
NET 12 FL. OZ. (355 mL)
cola
12 FL. OZ. 355 ml
quality guaranteed
Cola
NO FRILLS
NET 12 FL. OZ. (355ml)

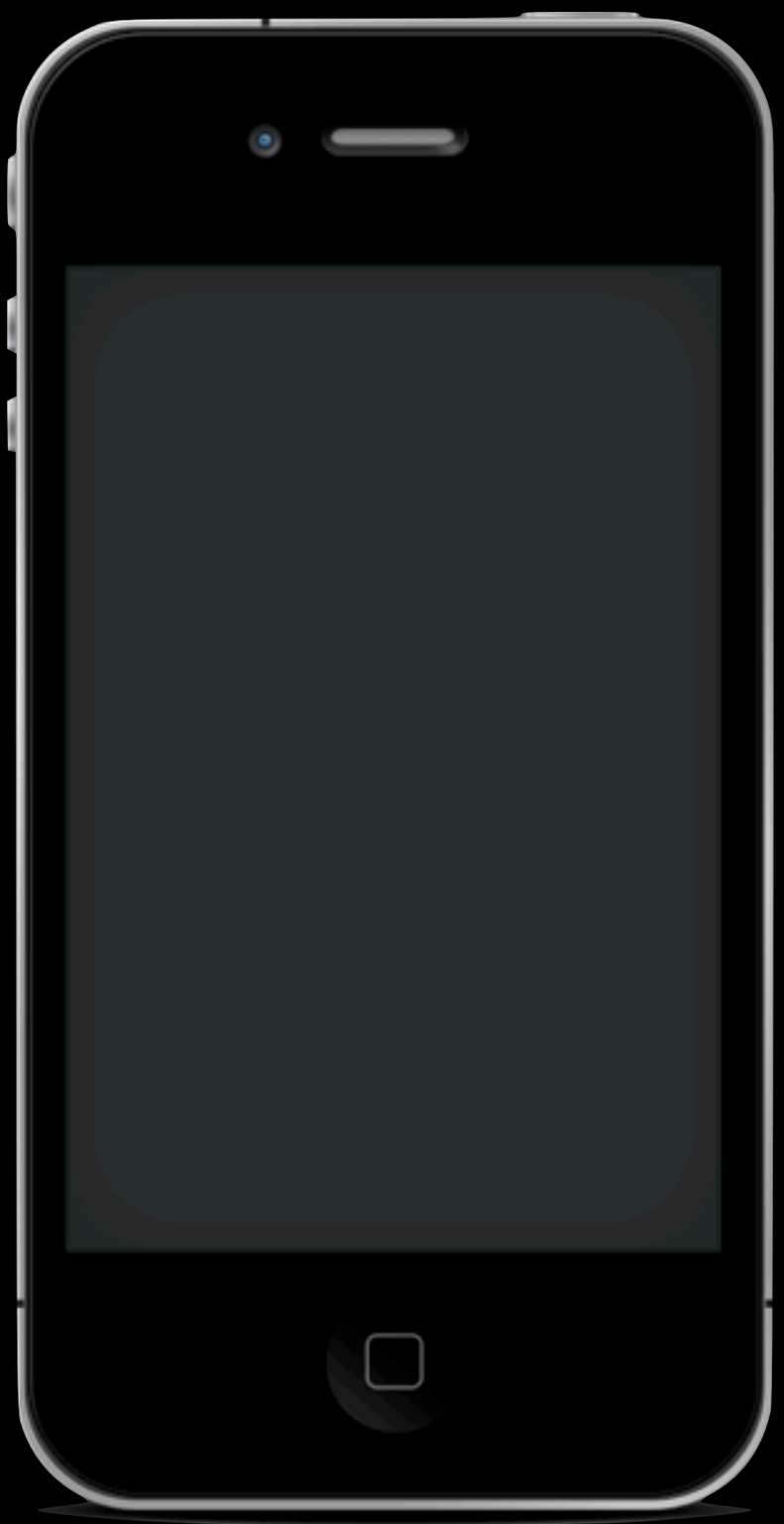
Let’s Build a Mobile App

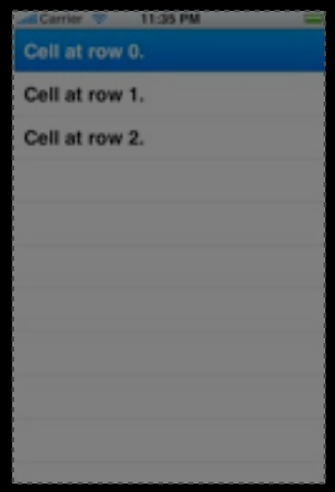


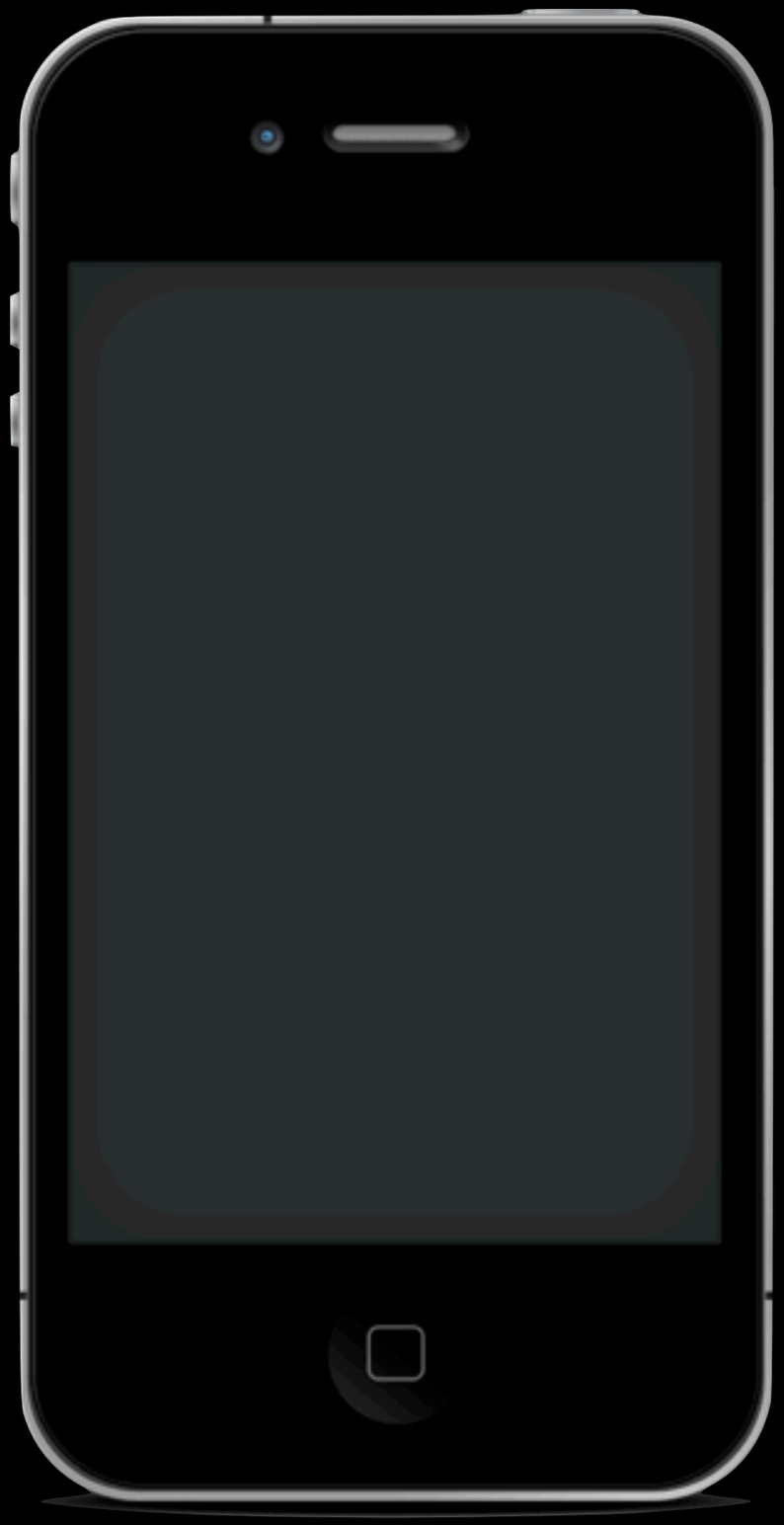

Carrier
11:35 PM
Cell at row 0.
Cell at row 1.
Cell at row 2.


Carrier
11:35 PM
Cell at row 0.
Cell at row 1.
Cell at row 2.

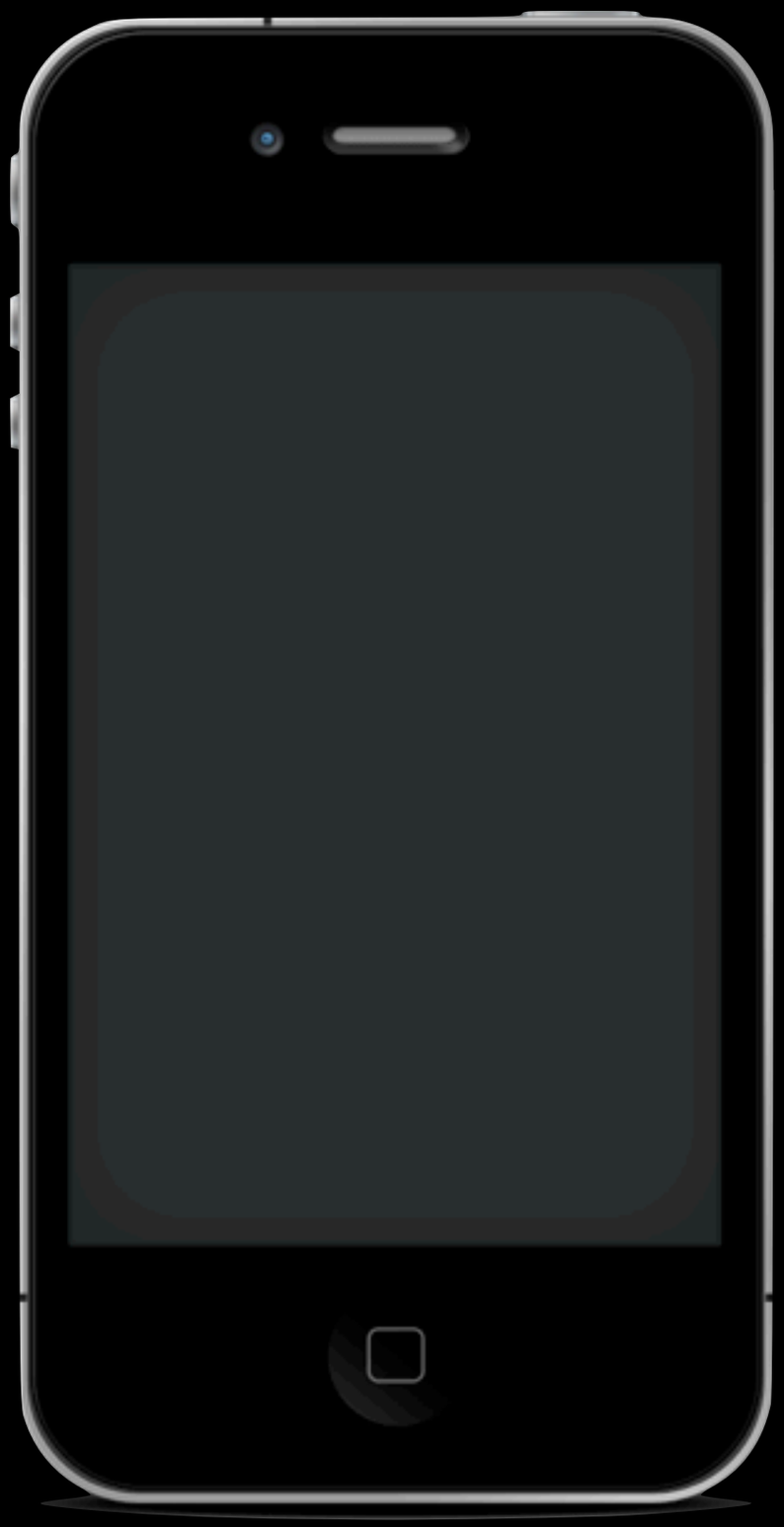

Carrier
11:35 PM
Cell at row 0.
Cell at row 1.
Cell at row 2.

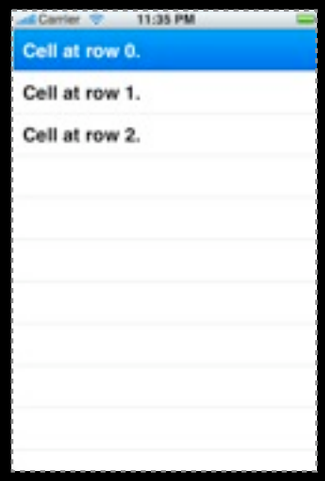



Carrier
11:35 PM
Cell at row 0.
Cell at row 1.
Cell at row 2.




Carrier
11:35 PM
Cell at row 0.
Cell at row 1.
Cell at row 2.

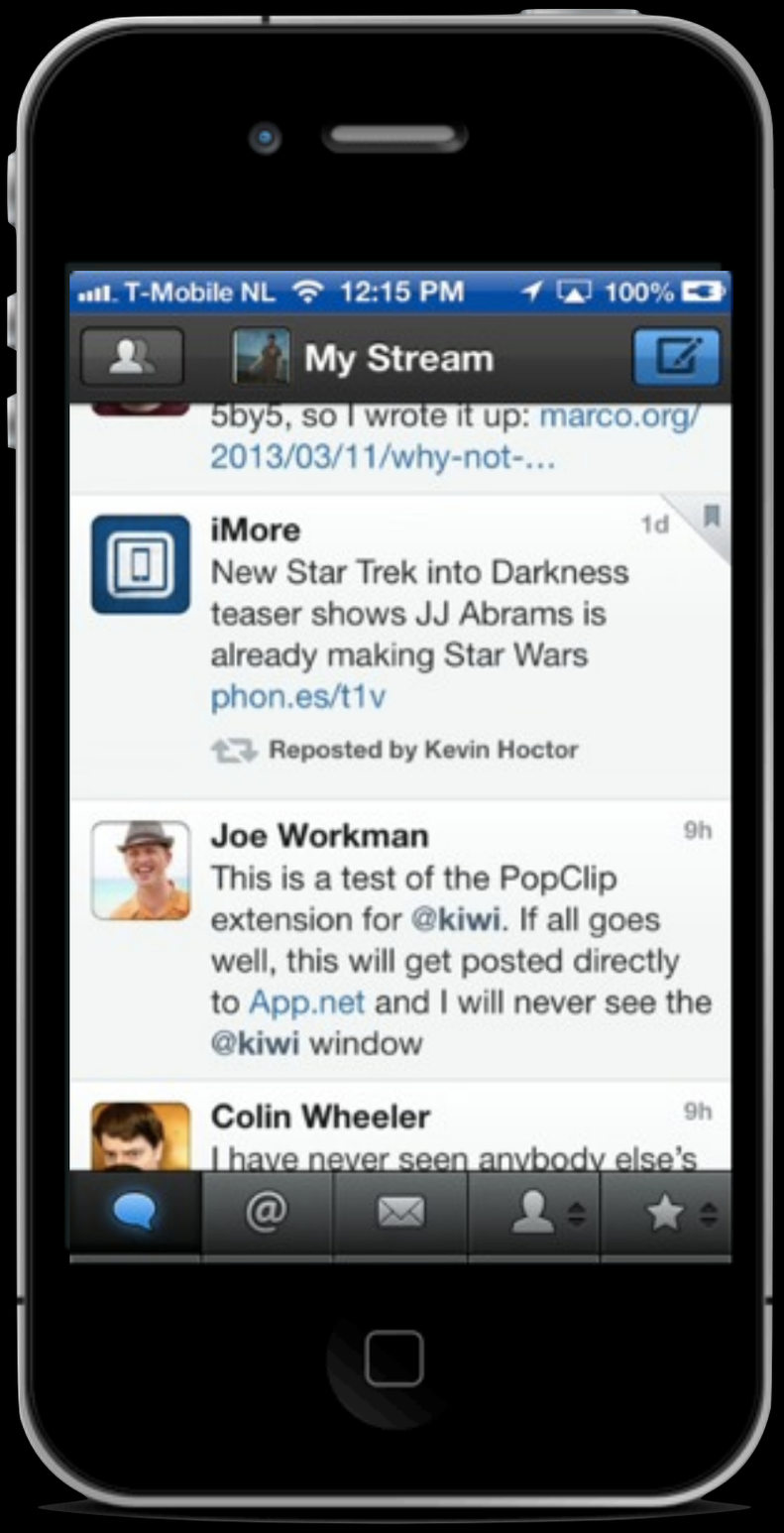




View

Controller

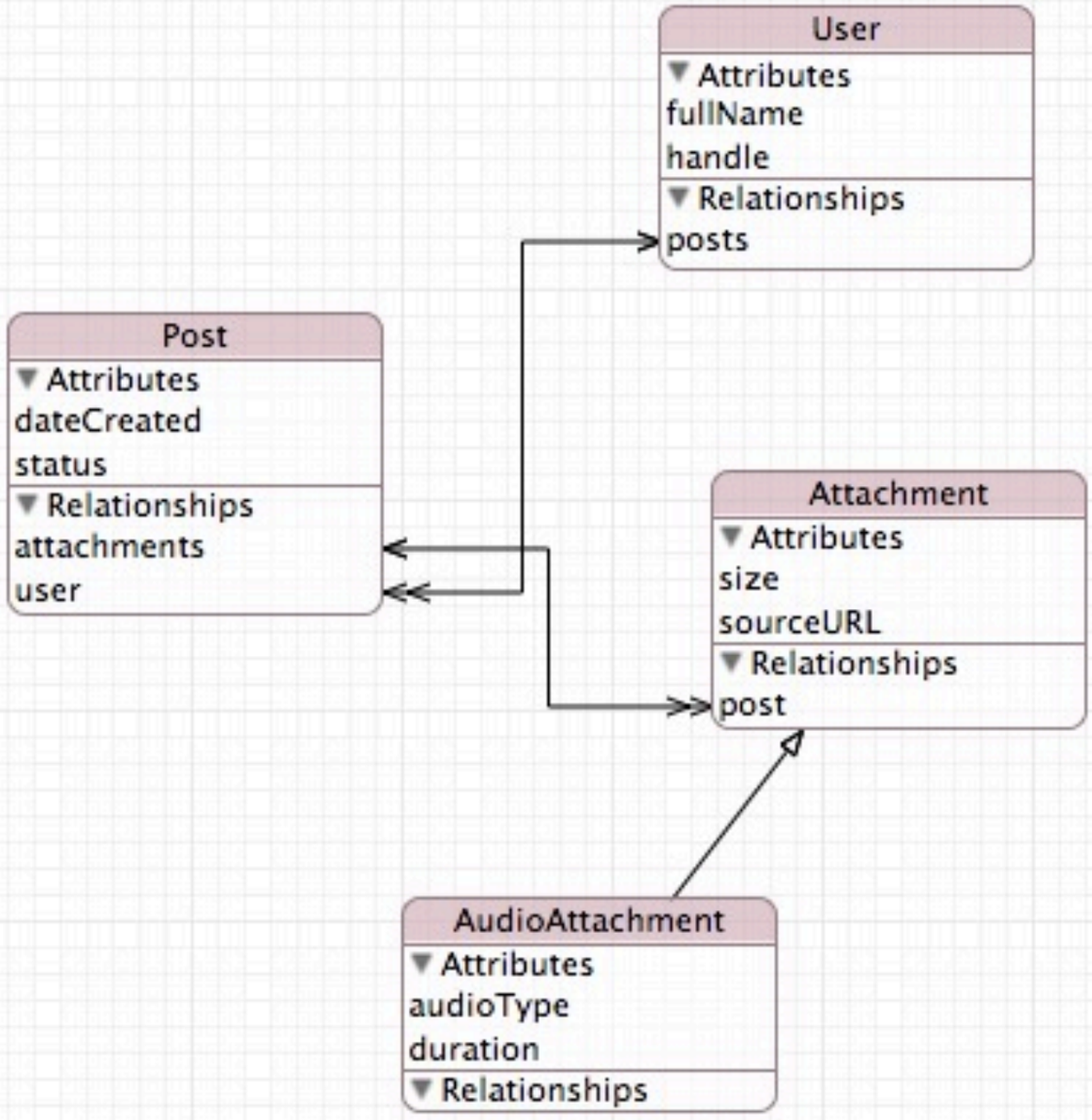
Model

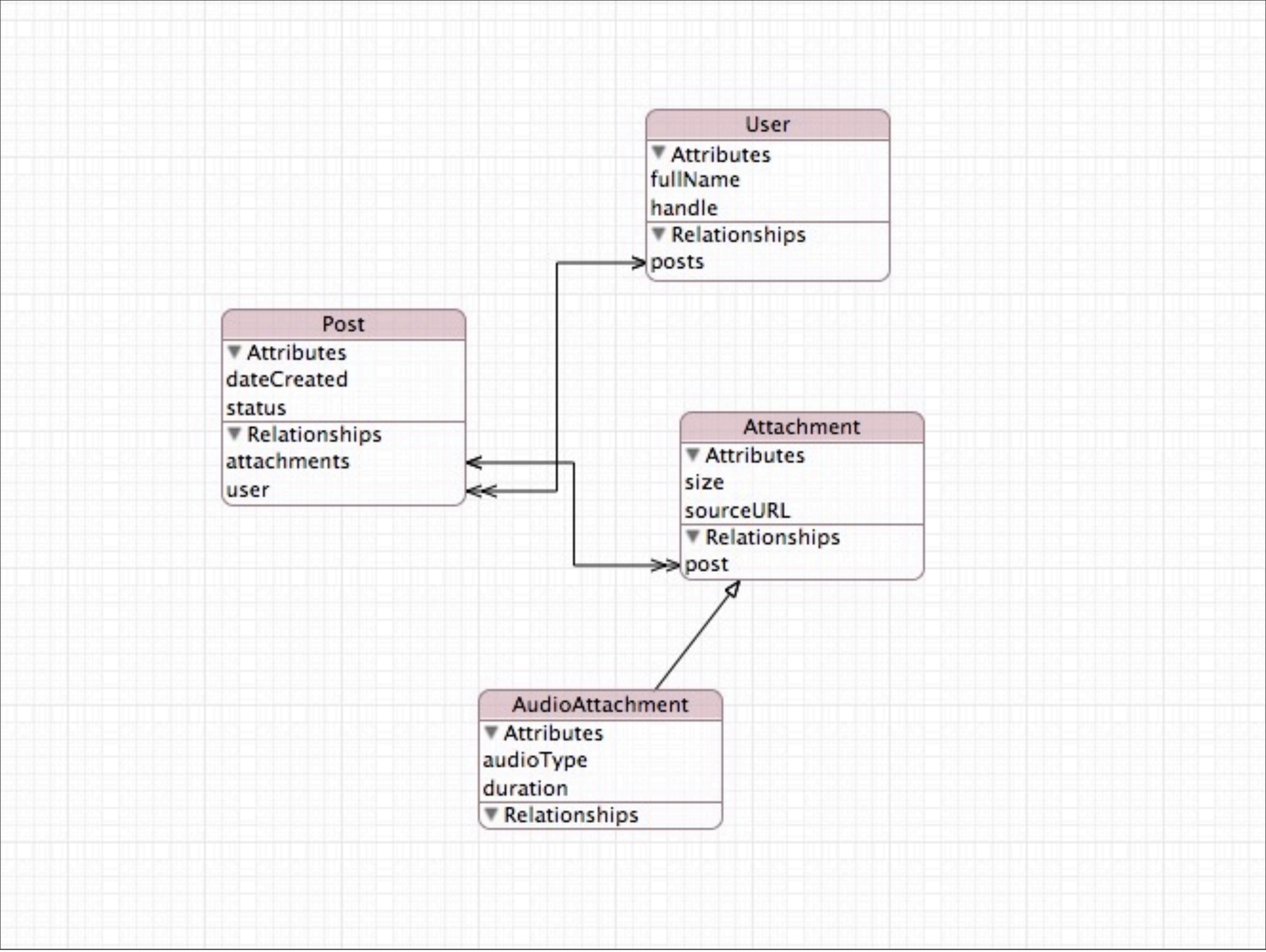

T-Mobile NL
12:15 PM
100%
My Stream
5by5, so I wrote it up: marco.org/
2013/03/11/why-not-…
iMore
1d
New Star Trek into Darkness
teaser shows JJ Abrams is
already making Star Wars
phon.es/t1v
Reposted by Kevin Hoctor
Joe Workman
9h
This is a test of the PopClip
extension for @kiwi. If all goes
well, this will get posted directly
to App.net and I will never see the
@kiwi window
Colin Wheeler
9h

MVC

Model View Controller

Model View Controller


User
Attributes
fullName
handle
Relationships
posts
Post
Attributes
dateCreated
status
Relationships
attachments
user
Attachment
Attributes
size
sourceURL
Relationships
post
AudioAttachment
Attributes
audioType
duration
Relationships


User
Attributes
fullName
handle
Relationships
posts
Post
Attributes
dateCreated
status
Relationships
attachments
user
Attachment
Attributes
size
sourceURL
Relationships
post
AudioAttachment
Attributes
audioType
duration
Relationships

```objc
@interface Post : _Post

+ (id) postWithId:(NSNumber *)postId;

- (void) downloadAttachments;

@end
```

Model View Controller

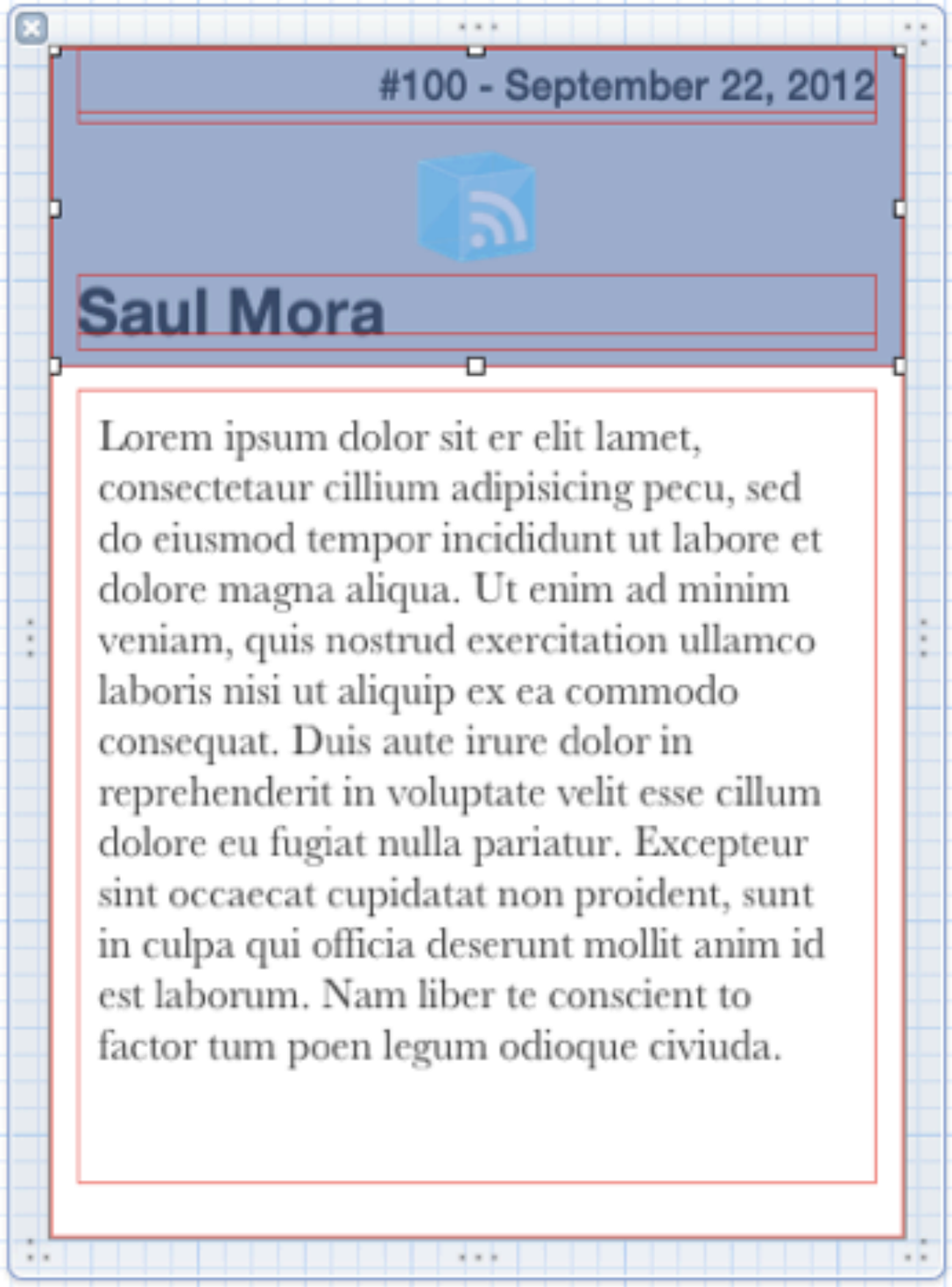
Placeholders
File's Owner
First Responder
Objects
View
Image View – NSBrief_logo.png
Text View
Label – Saul Mora
Label – #100 – September...
#100 - September 22, 2012
Saul Mora
Lorem ipsum dolor sit er elit lamet, consectetaur cillium adipisicing pecu, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum. Nam liber te conscient to factor tum poen legum odioque civiuda.

Placeholders

- File's Owner
- First Responder

Objects

- View
  - Image View – NSBrief_logo.png
  - Text View
  - Label – Saul Mora
  - Label – #100 – September…



Model View Controller

?

!

Twitter / Colin_Campbell: iOS architecture, where MVC ...

Twitter, Inc. twitter.com/Colin_Campbell/status/293167951132098560

Reader

Intel EPP Apple Store | my pinboard | Pin It | Task ✔ | Ping Feedburner (NSBrief) | Add to Snippets | Read Later

Twitter / Colin_Campbell: iOS architecture, where MVC ...

Search

Have an account? **Sign In**

Colin Campbell
@Colin_Campbell

Follow

iOS architecture, where MVC stands for Massive View Controller

Reply | Retweet | Favorite | More

193 RETWEETS

69 FAVORITES

5:27 PM - Jan 20, 2013

Jesse Armand @jessearmand — Jan 21
@Colin_Campbell Correct.
Details

Fredrik Olsson @PeyloW — Jan 21
@Colin_Campbell @Jussi7 Because people are too afraid to let their FOOSongView view class know about their FOOSong model class :(.
Details

Favstar.fm 50★'s @favstar50 — Jan 21
@Colin_Campbell Congrats on your 50★ tweet!
favstar.fm/t/293167951132...

Search

Have an

Colin Campbell
@Colin_Campbell

Follow

iOS architecture, where MVC stands for Massive View Controller

Reply Retweet Favorite More

193 RETWEETS

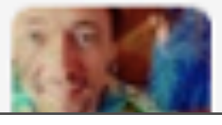
69 FAVORITES

5:27 PM - Jan 20, 2013

Jesse Armand @jessearmand Jan 21

@Colin_Campbell Correct.

Details



Fredrik Olsson @PeyloW Jan 21

@Colin_Campbell @Jussi7 Because people are too afraid to

View

ViewController

Model

View

View

View

ViewController

ViewController

ViewController

Model

View

ViewController

Model

MVC can be composed of MVC (UITextView for example)

View

ViewController

Model

MVC can be composed of MVC (UITextView for example)

Network

View

App.net

ViewController

Visual Control

Model

Database

ViewController

Network

App.net

Visual Control

Database

ViewController

Network

App.net

Visual Control

Database


T-Mobile NL
12:15 PM
100%
My Stream
5by5, so I wrote it up: marco.org/2013/03/11/why-not-…
iMore
1d
New Star Trek into Darkness teaser shows JJ Abrams is already making Star Wars phon.es/t1v
Reposted by Kevin Hoctor
Joe Workman
9h
This is a test of the PopClip extension for @kiwi. If all goes well, this will get posted directly to App.net and I will never see the @kiwi window
Colin Wheeler
9h
I have never seen anybody else's

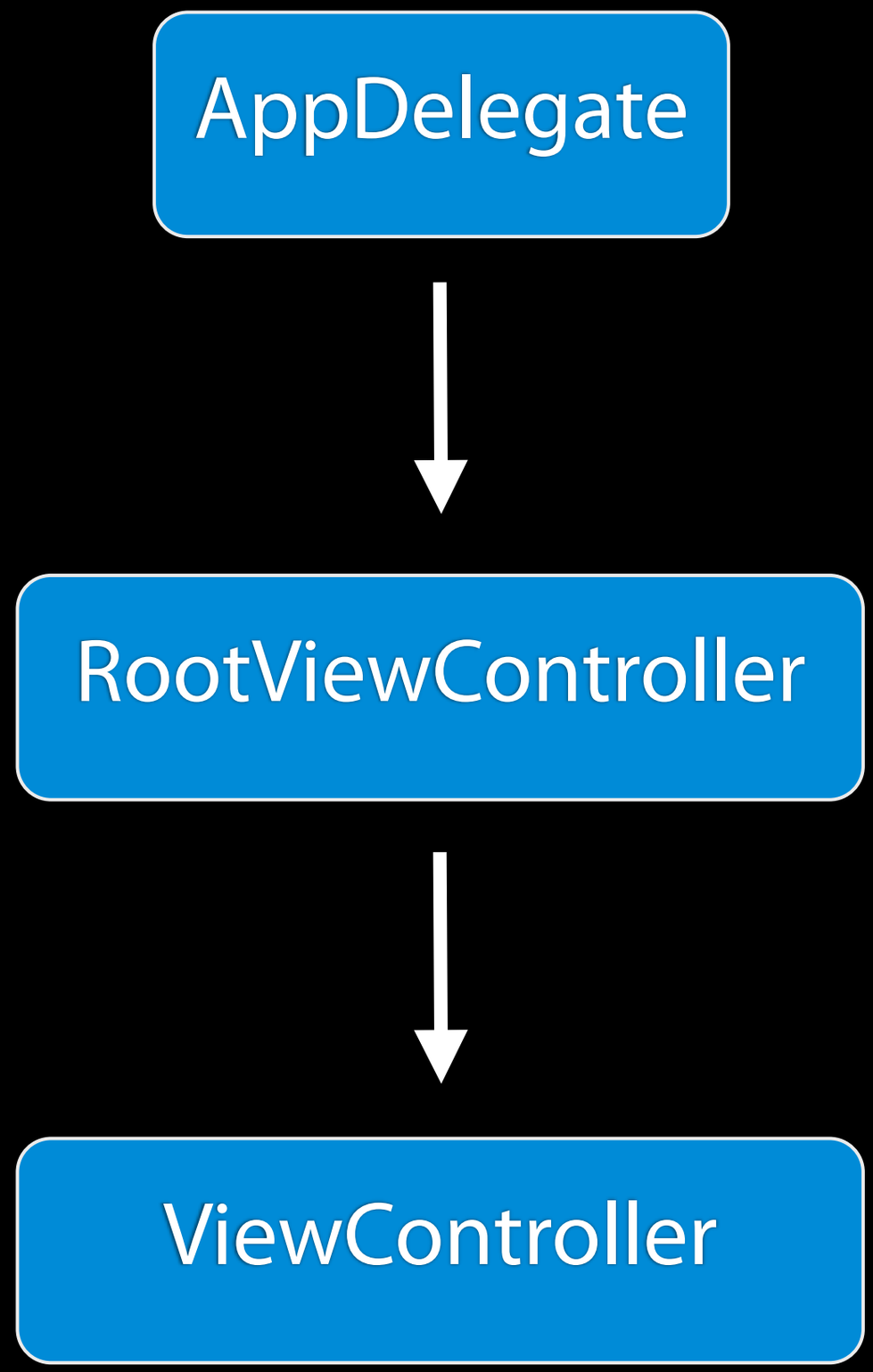
AppDelegate

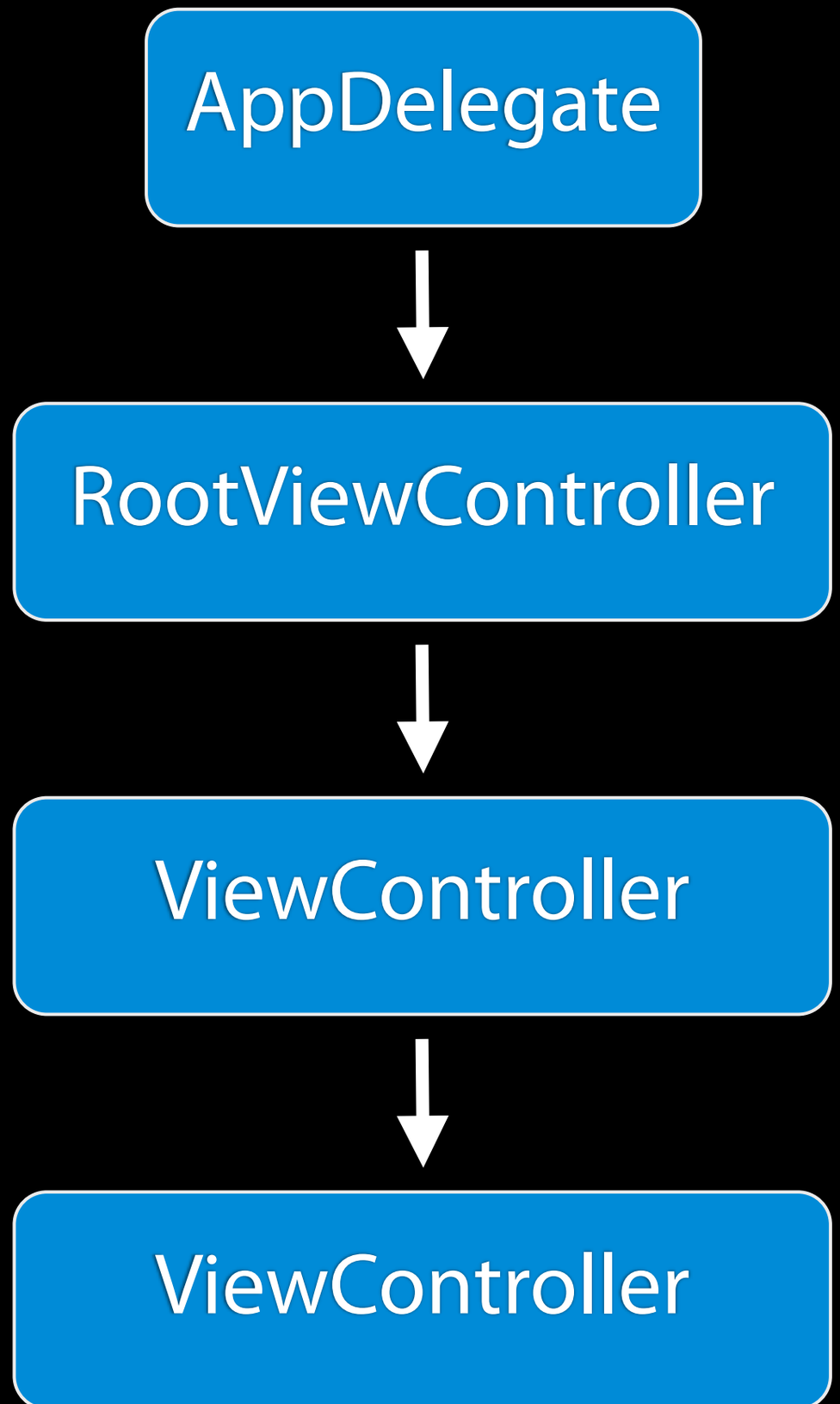
AppDelegate

RootViewController

AppDelegate

RootViewController


AppDelegate
RootViewController
ViewController


AppDelegate
RootViewController
ViewController
ViewController

AppDelegate

RootViewController

ViewController

ViewController

ViewController

ViewController

AppDelegate

RootViewController

ViewController

ViewController

ViewController

ViewController

ViewController

ViewController

ViewController

AppDelegate

RootViewController

ViewController

ViewController

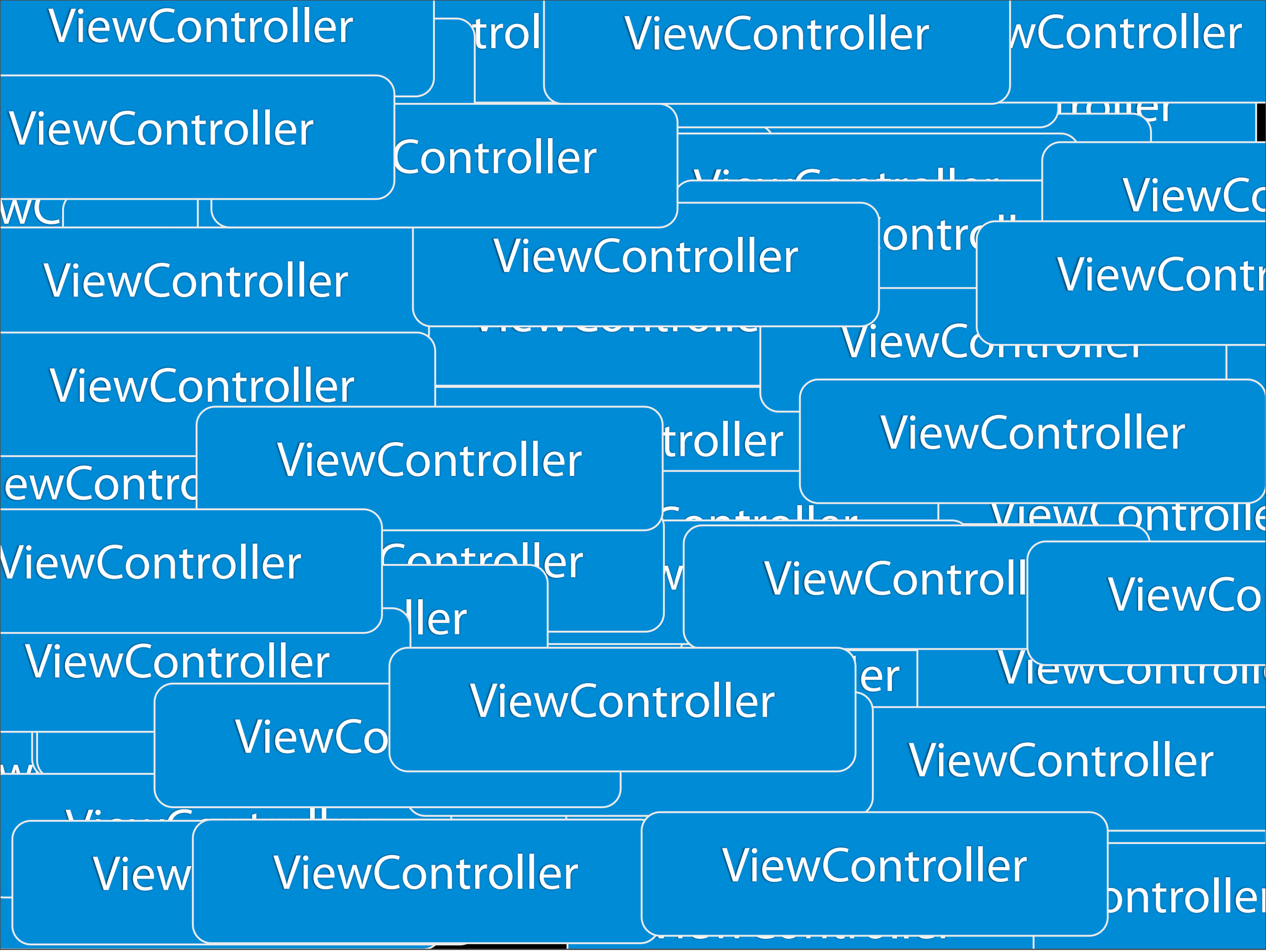
ViewController

ViewController

ViewController

ViewController

ViewController


ViewController
ViewController
ViewController
ViewController
ViewController
ViewController
ViewController
ViewController
ViewController
ViewController
ViewController
ViewController
ViewController
ViewController
ViewController
ViewController
ViewController
ViewController
ViewController
ViewController

AppDelegate

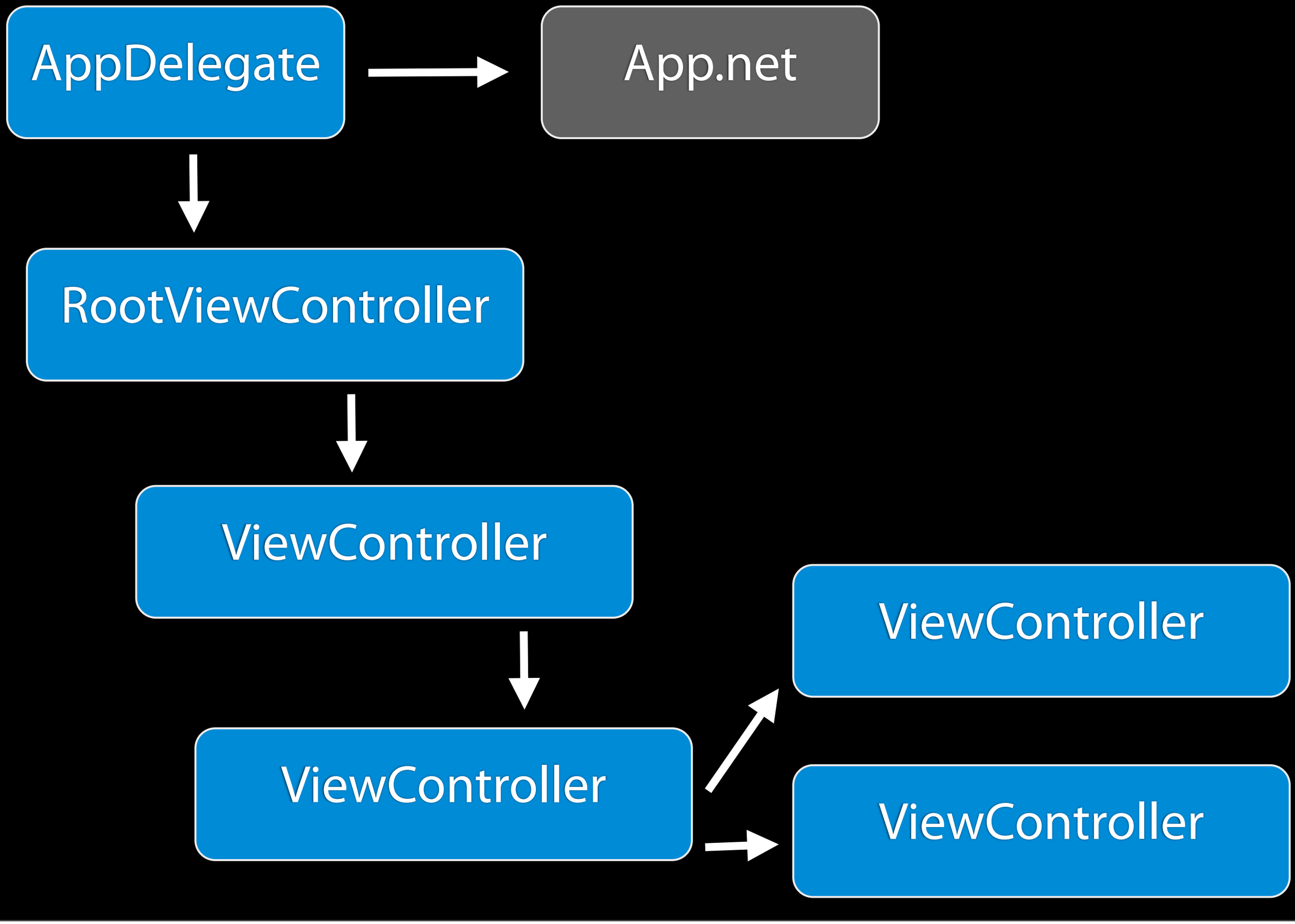
RootViewController

ViewController

ViewController

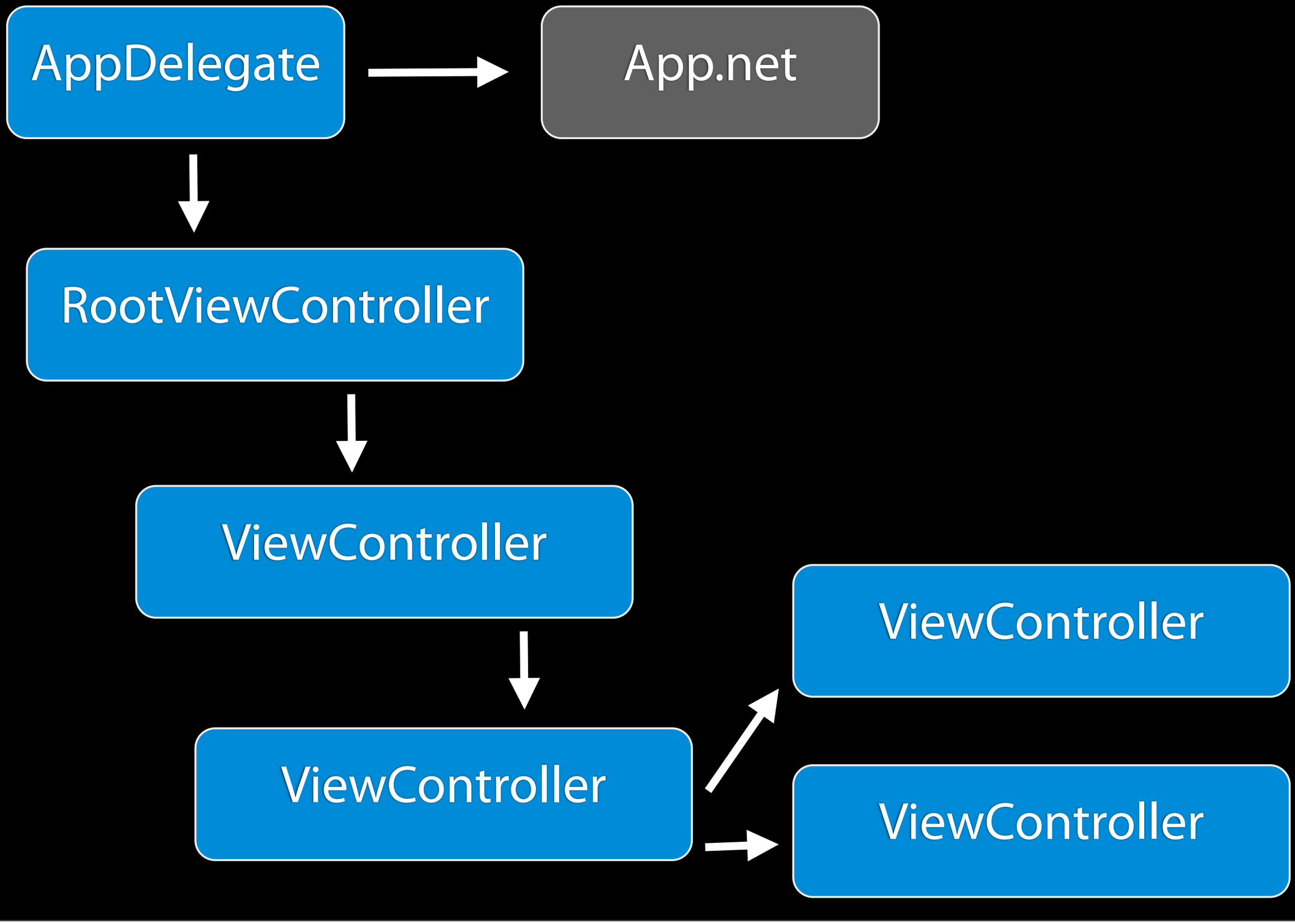
ViewController

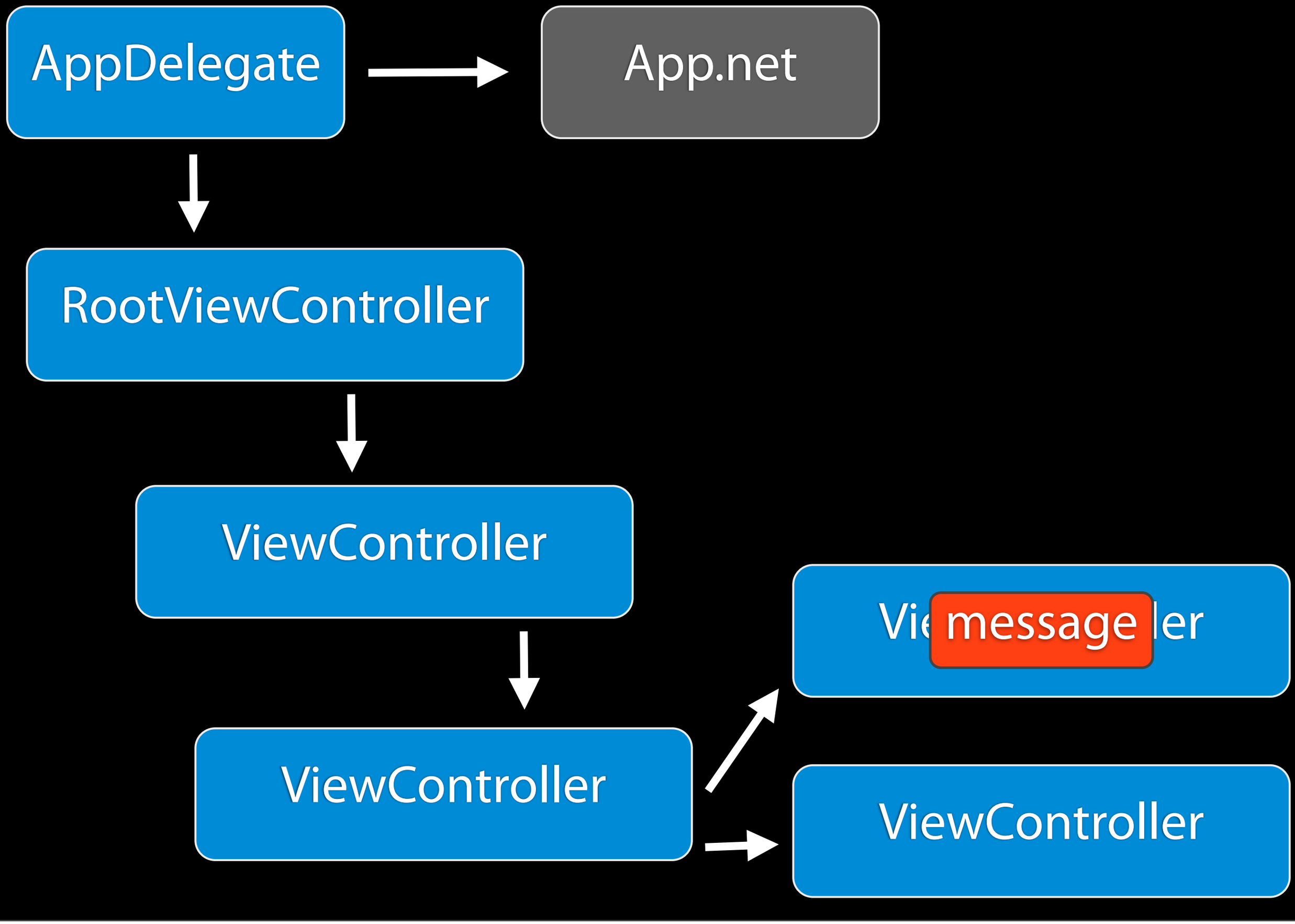
ViewController

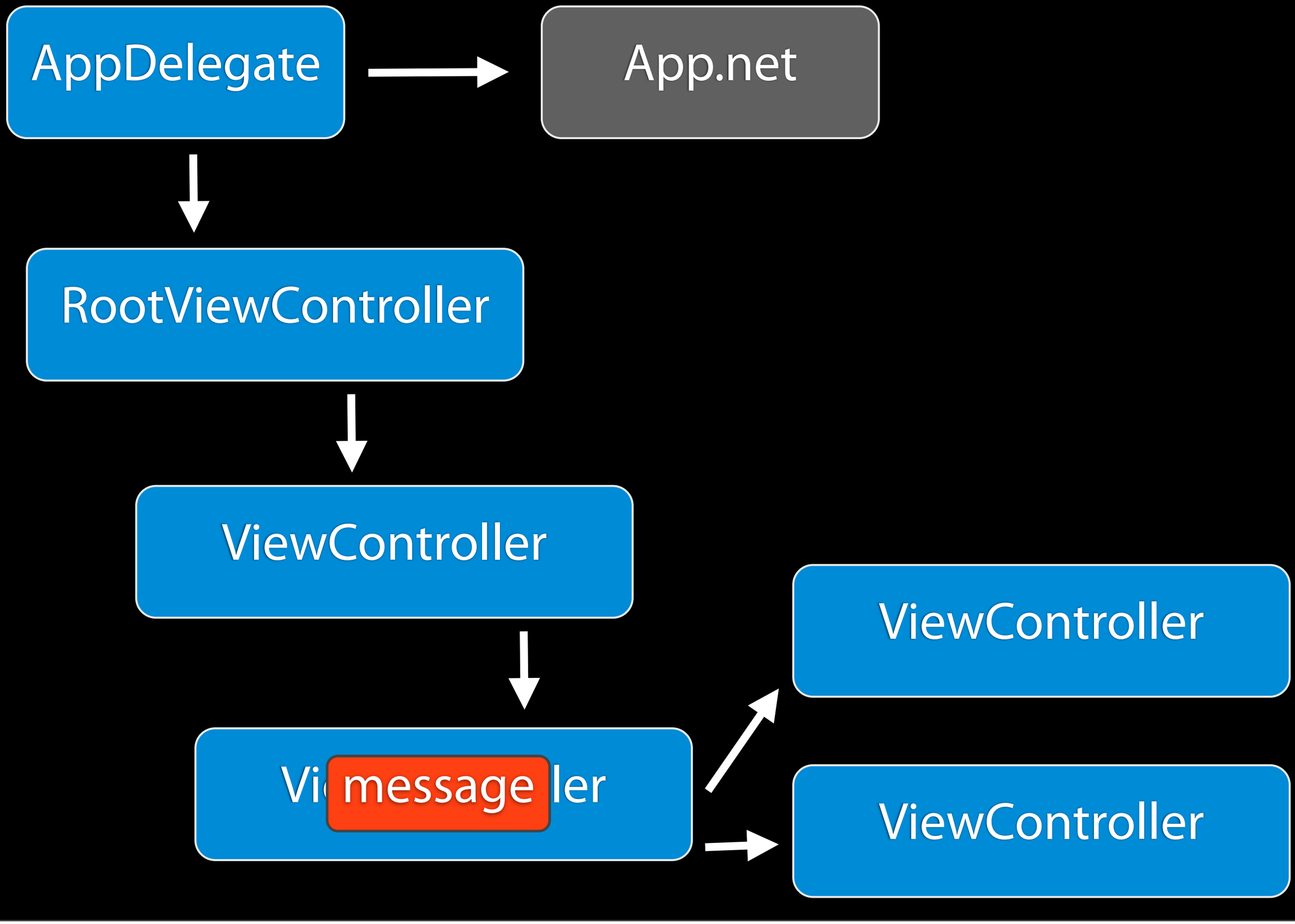

AppDelegate
App.net
RootViewController
ViewController
ViewController
ViewController
ViewController

```
(MyAppDelegate *)[[UIApplication
sharedApplication] delegate]
```

```
#define sharedDelegate \
(MyAppDelegate *)[[UIApplication sharedApplication] delegate]
```


AppDelegate
App.net
RootViewController
ViewController
ViewController
ViewController
ViewController


AppDelegate
App.net
RootViewController
ViewController
ViewController
message
ViewController


AppDelegate
App.net
RootViewController
ViewController
Vi message ler
ViewController
ViewController

AppDelegate

App.net

RootViewController

Vi message ller

ViewController

ViewController

ViewController

ViewController

AppDelegate

App.net

Root message roller

ViewController

ViewController

ViewController

ViewController

A message te

App.net

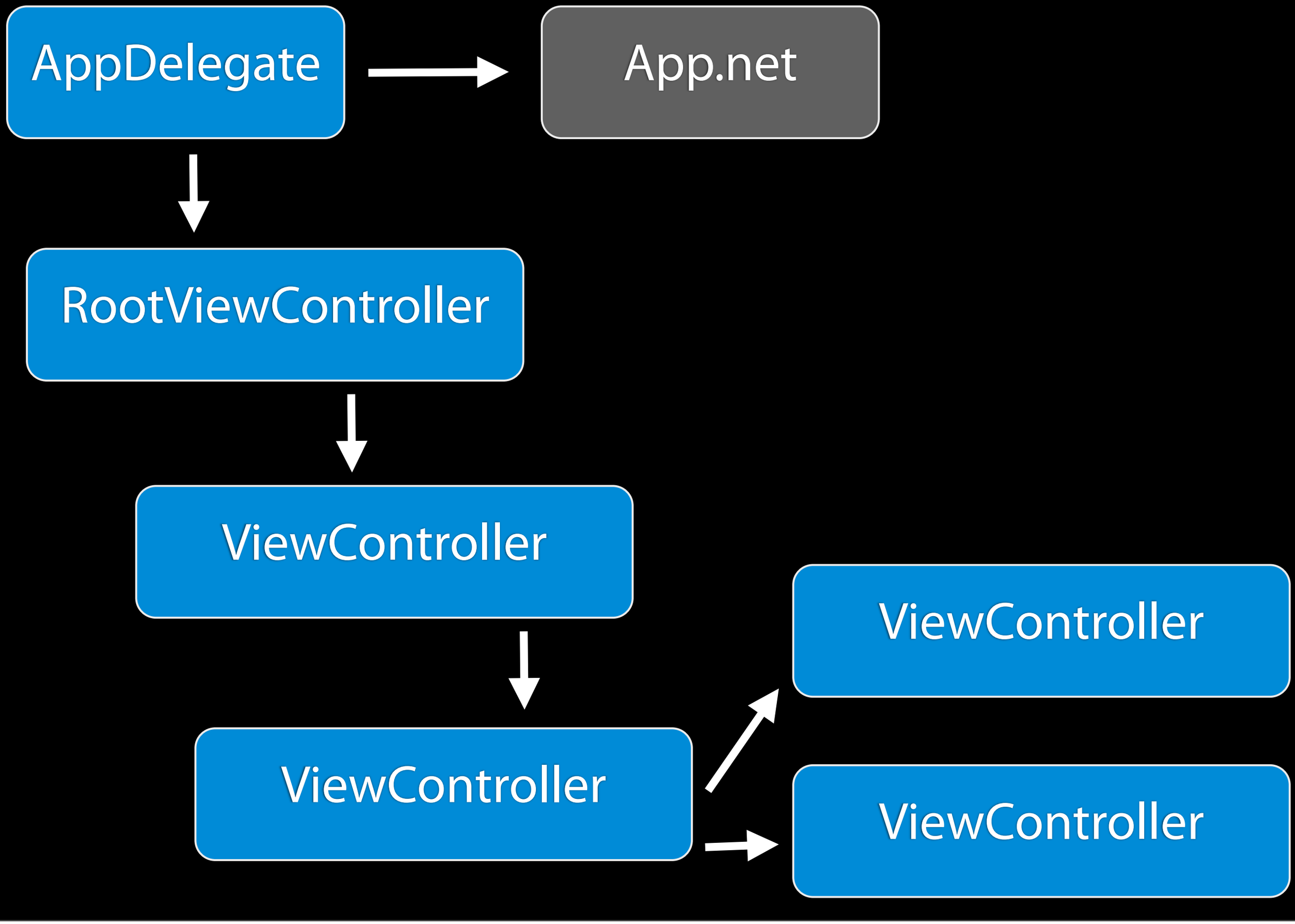
RootViewController

ViewController

ViewController

ViewController

ViewController


AppDelegate
App.net
RootViewController
ViewController
ViewController
ViewController
ViewController

Pause here more, explain more :

What
When to use
Why

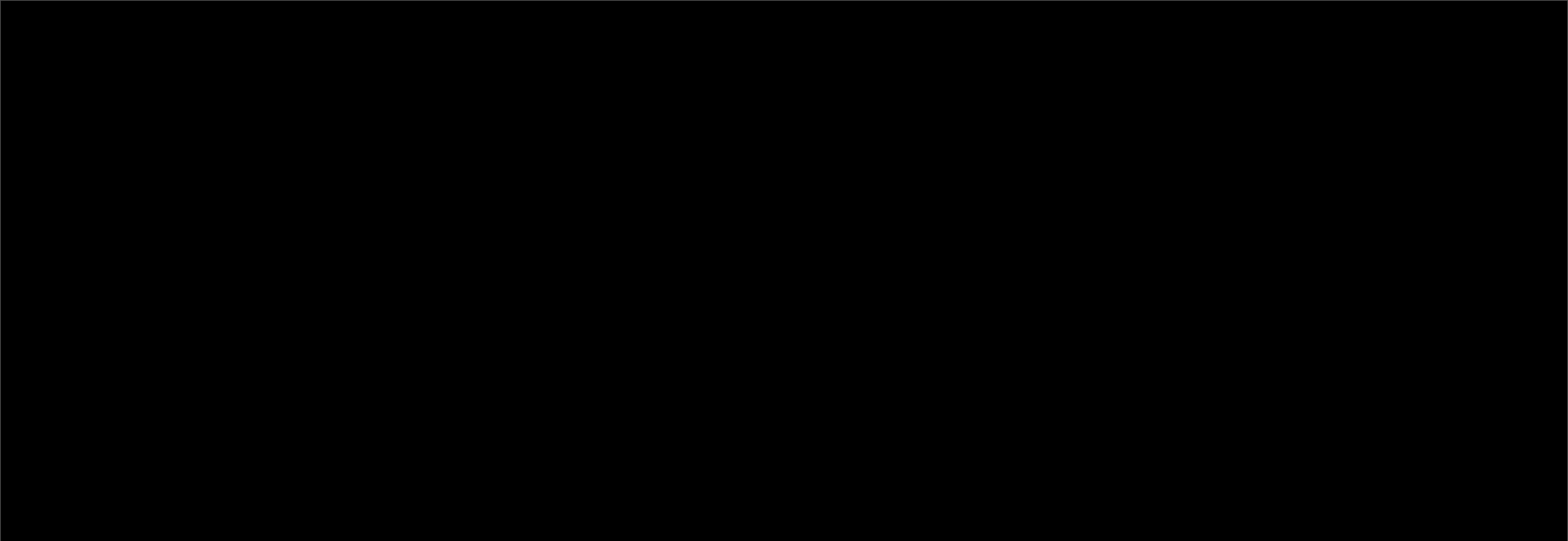
Chain of Responsibility

Responder Chain

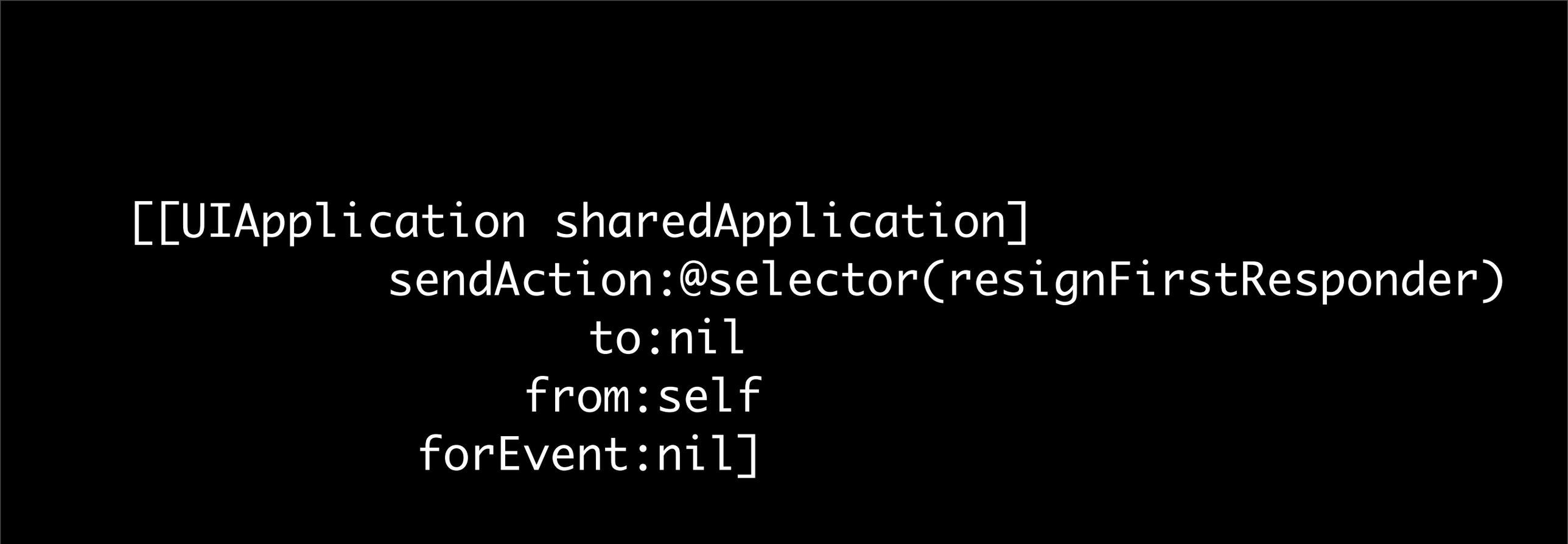
```
[[UIApplication
sharedApplication]
sendAction:to:from:forEvent:]
```

```
@implementation UIView
(FindAndResignFirstResponder)

- (BOOL)findAndResignFirstResponder
{
    if ([self isFirstResponder]) {
        [self resignFirstResponder];
        return YES;
    }
    for (UIView *subView in [self subviews]) {
        if ([subView findAndResignFirstResponder])
            return YES;
    }
    return NO;
}
@end
```

```
while ([obj retainCount]) [obj release];
```


Q W E R T Y U I O P
A S D F G H J K L return
Z X C V B N M ! , ? .
.?123
.?123

```
[[UIApplication sharedApplication]
          sendAction:@selector(resignFirstResponder)
                  to:nil
                from:self
           forEvent:nil]
```



```
[[UIApplication sharedApplication]
          sendAction:@selector(resignFirstResponder)
                  to:nil
                from:self
           forEvent:nil]
```

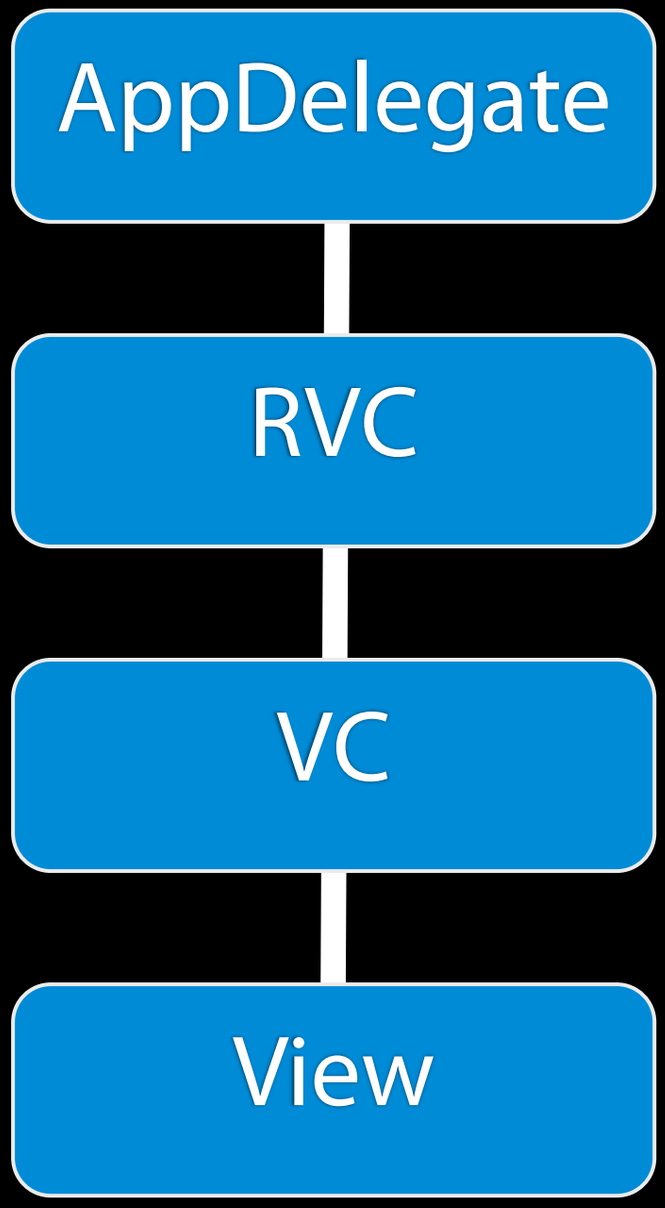
- (void)viewDidAppear:

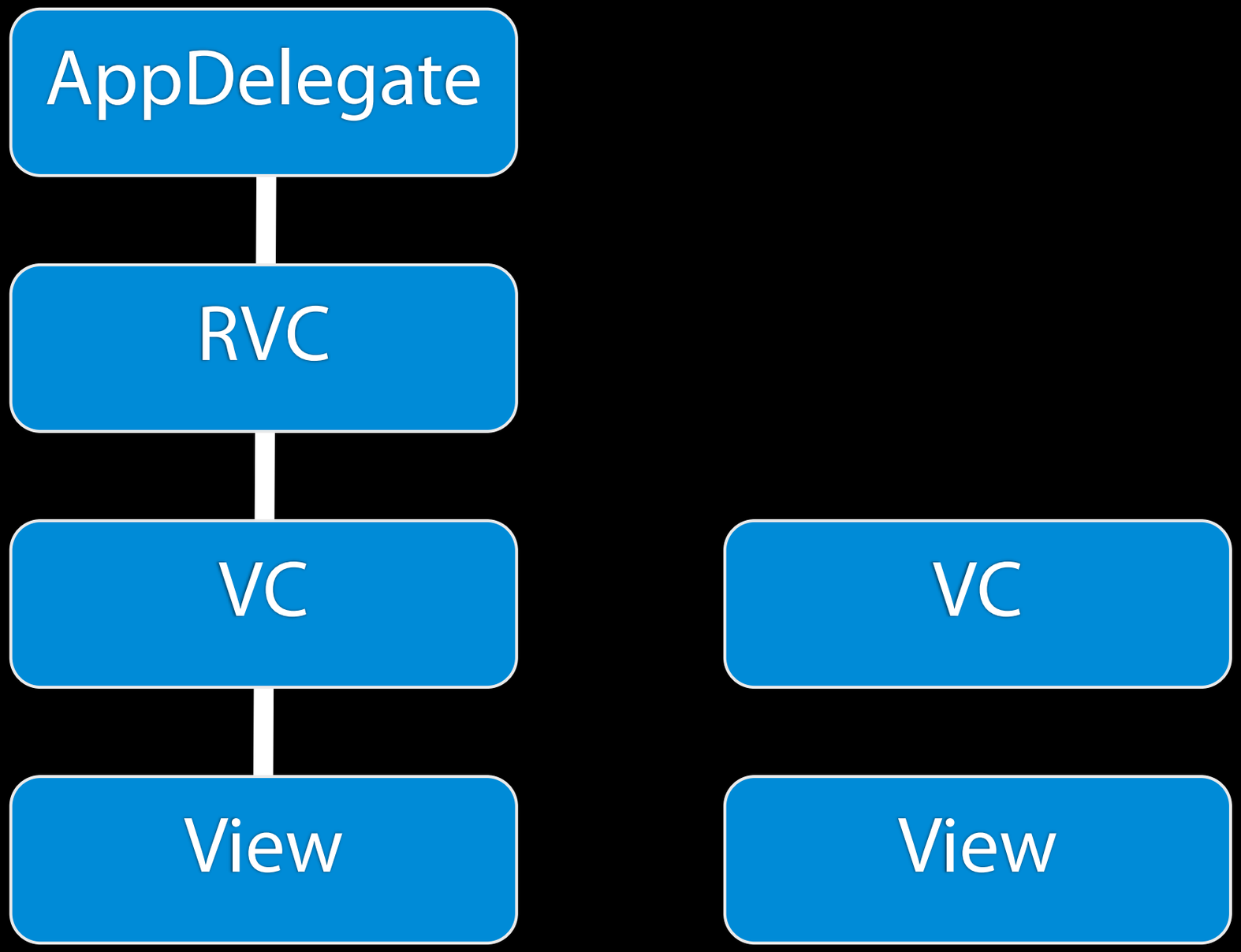
AppDelegate

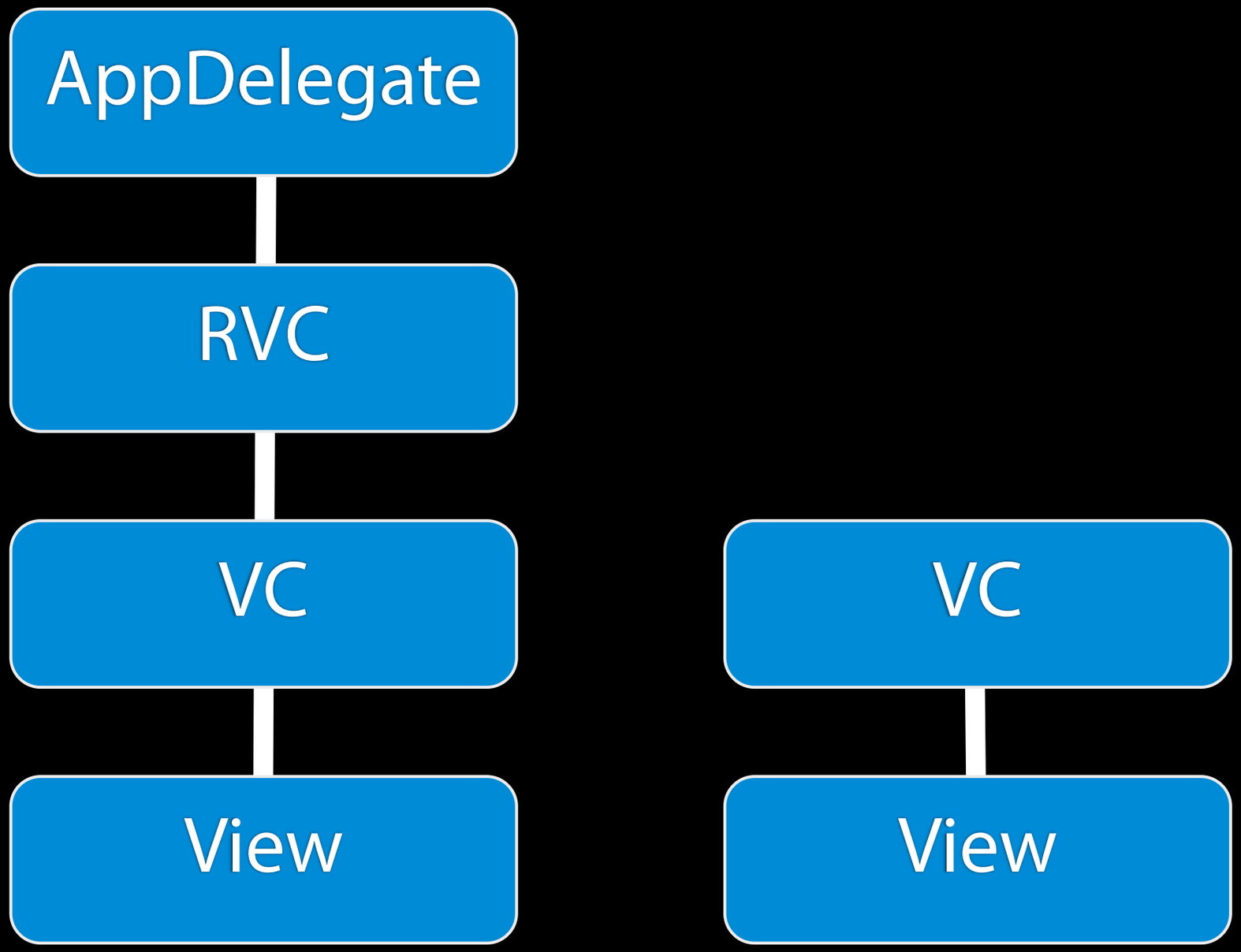
RVC

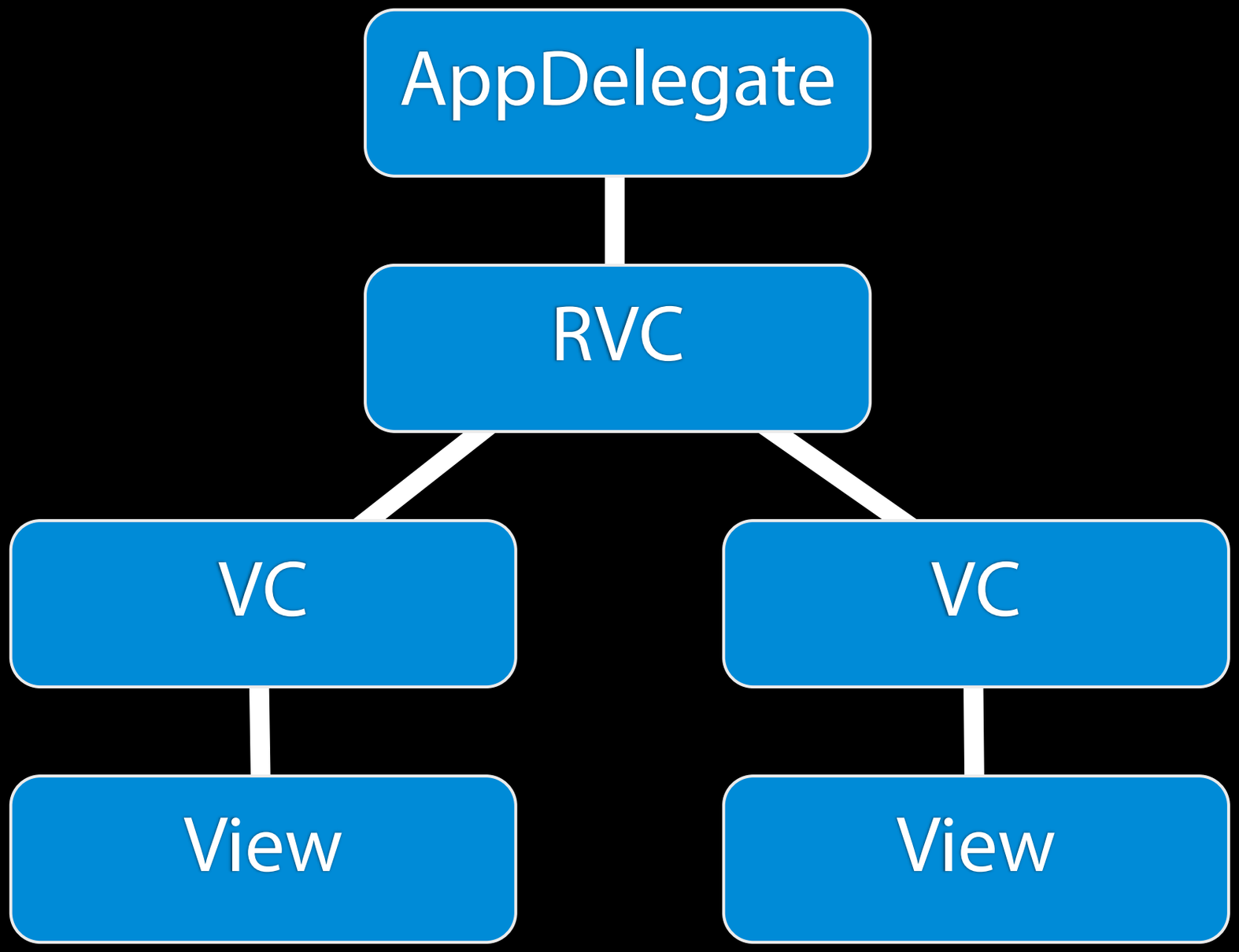
VC

View


AppDelegate
RVC
VC
View


AppDelegate
RVC
VC
View
VC
View


AppDelegate
RVC
VC
View
VC
View


AppDelegate
RVC
VC
VC
View
View


AFNetworking

```objc
- (void)viewDidLoad {
    [super viewDidLoad];

    // Setting Up Table View
    self.tableView = [[UITableView alloc] initWithFrame:CGRectMake(0.0, 0.0, self.view.bounds.size.width,
self.view.bounds.size.height) style:UITableViewStylePlain];
    self.tableView.dataSource = self;
    self.tableView.delegate = self;
    self.tableView.autoresizingMask = UIViewAutoresizingFlexibleWidth | UIViewAutoresizingFlexibleHeight;
    self.tableView.hidden = YES;
    [self.view addSubview:self.tableView];

    // Setting Up Activity Indicator View
    self.activityIndicatorView = [[UIActivityIndicatorView alloc]
initWithActivityIndicatorStyle:UIActivityIndicatorViewStyleGray];
    self.activityIndicatorView.hidesWhenStopped = YES;
    self.activityIndicatorView.center = self.view.center;
    [self.view addSubview:self.activityIndicatorView];
    [self.activityIndicatorView startAnimating];

    // Initializing Data Source
    self.movies = [[NSArray alloc] init];

    NSURL *url = [[NSURL alloc] initWithString:@"http://itunes.apple.com/search?term=harry&country=us&entity=movie"];
    NSURLRequest *request = [[NSURLRequest alloc] initWithURL:url];

    AFJSONRequestOperation *operation = [AFJSONRequestOperation JSONRequestOperationWithRequest:request
success:^(NSURLRequest *request, NSHTTPURLResponse *response, id JSON) {
        self.movies = [JSON objectForKey:@"results"];
        [self.activityIndicatorView stopAnimating];
        [self.tableView setHidden:NO];
        [self.tableView reloadData];

    } failure:^(NSURLRequest *request, NSHTTPURLResponse *response, NSError *error, id JSON) {
        NSLog(@"Request Failed with Error: %@, %@", error, error.userInfo);
    }];

    [operation start];
}
```

```objc
- (void)viewDidLoad {
    [super viewDidLoad];

    // Setting Up Table View
    self.tableView = [[UITableView alloc] initWithFrame:CGRectMake(0.0, 0.0, self.view.bounds.size.width,
self.view.bounds.size.height) style:UITableViewStylePlain];
    self.tableView.dataSource = self;
    self.tableView.delegate = self;
    self.tableView.autoresizingMask = UIViewAutoresizingFlexibleWidth | UIViewAutoresizingFlexibleHeight;
    self.tableView.hidden = YES;
    [self.view addSubview:self.tableView];

    // Setting Up Activity Indicator View
    self.activityIndicatorView = [[UIActivityIndicatorView alloc]
initWithActivityIndicatorStyle:UIActivityIndicatorViewStyleGray];
    self.activityIndicatorView.hidesWhenStopped = YES;
    self.activityIndicatorView.center = self.view.center;
    [self.view addSubview:self.activityIndicatorView];
    [self.activityIndicatorView startAnimating];

    // Initializing Data Source
    self.movies = [[NSArray alloc] init];

    NSURL *url = [[NSURL alloc] initWithString:@"http://itunes.apple.com/search?term=harry&country=us&entity=movie"];
    NSURLRequest *request = [[NSURLRequest alloc] initWithURL:url];

    AFJSONRequestOperation *operation = [AFJSONRequestOperation JSONRequestOperationWithRequest:request
success:^(NSURLRequest *request, NSHTTPURLResponse *response, id JSON) {
        self.movies = [JSON objectForKey:@"results"];
        [self.activityIndicatorView stopAnimating];
        [self.tableView setHidden:NO];
        [self.tableView reloadData];

    } failure:^(NSURLRequest *request, NSHTTPURLResponse *response, NSError *error, id JSON) {
        NSLog(@"Request Failed with Error: %@, %@", error, error.userInfo);
    }];

    [operation start];
}
```

```
    // Initializing Data Source
    self.movies = [[NSArray alloc] init];

    NSURL *url = [[NSURL alloc] initWithString:@"http://itunes.apple.com/search?
term=harry&country=us&entity=movie"];
    NSURLRequest *request = [[NSURLRequest alloc] initWithURL:url];

    AFJSONRequestOperation *operation = [AFJSONRequestOperation
JSONRequestOperationWithRequest:request success:^(NSURLRequest *request,
NSHTTPURLResponse *response, id JSON) {
        self.movies = [JSON objectForKey:@"results"];
        [self.activityIndicatorView stopAnimating];
        [self.tableView setHidden:NO];
        [self.tableView reloadData];

    } failure:^(NSURLRequest *request, NSHTTPURLResponse *response, NSError
*error, id JSON) {
        NSLog(@"Request Failed with Error: %@, %@", error, error.userInfo);
    }];

    [operation start];
```

```
    // Initializing Data Source
    self.movies = [[NSArray alloc] init];

    NSURL *url = [[NSURL alloc] initWithString:@"http://itunes.apple.com/search?
term=harry&country=us&entity=movie"];
    NSURLRequest *request = [[NSURLRequest alloc] initWithURL:url];

    AFJSONRequestOperation *operation = [AFJSONRequestOperation
JSONRequestOperationWithRequest:request success:^(NSURLRequest *request,
NSHTTPURLResponse *response, id JSON) {
        self.movies = [JSON objectForKey:@"results"];
        [self.activityIndicatorView stopAnimating];
        [self.tableView setHidden:NO];
        [self.tableView reloadData];

    } failure:^(NSURLRequest *request, NSHTTPURLResponse *response, NSError
*error, id JSON) {
        NSLog(@"Request Failed with Error: %@, %@", error, error.userInfo);
    }];

    [operation start];
```

```
^(NSURLRequest *request, NSHTTPURLResponse *response, id JSON) {
        self.movies = [JSON objectForKey:@"results"];
        [self.activityIndicatorView stopAnimating];
        [self.tableView setHidden:NO];
        [self.tableView reloadData];
    }
```

ADN Service

ADN Service

https://alpha-api.app.net/stream/0/posts/stream

ADN Service

https://alpha-api.app.net/stream/0/posts/stream

ADN Service

https://alpha-api.app.net/stream/0/posts/stream

POST:
include_reposters=1&include_annotations=1&incl
ude_muted=0&include_starred_by=1

ADN Service

Get Personal Stream

ADN Service

Get Personal Stream

include_reposters

include_muted

include_annotations

include_starred_by

ADN Service

Get Personal Stream

ADN Service

Get Personal Stream

ADN Service

Get Personal Stream

```
@property (nonatomic, assign) BOOL includeReposters

@property (nonatomic, assign) BOOL includeMuted

@property (nonatomic, assign) BOOL includeAnnotations

@property (nonatomic, assign) BOOL includeStarredBy
```

ADN Service

@class ADNRetrievePersonalStreamCommand : ADNCommand

@property (nonatomic, assign) BOOL includeReposters

@property (nonatomic, assign) BOOL includeMuted

@property (nonatomic, assign) BOOL includeAnnotations

@property (nonatomic, assign) BOOL includeStarredBy

@class ADNService : UIResponder

ADN Service

@property (nonatomic, copy) NSURL *baseURL;

@class ADNRetrievePersonalStreamCommand : ADNCommand

@property (nonatomic, assign) BOOL includeReposters

@property (nonatomic, assign) BOOL includeMuted

@property (nonatomic, assign) BOOL includeAnnotations

@property (nonatomic, assign) BOOL includeStarredBy

ADN Service

Get Personal Stream

ADN Service

Command

ADNCommand

ADNCommand

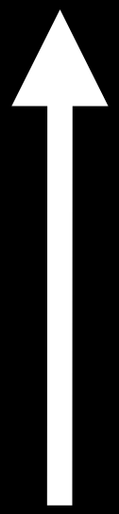
ADNCommand

ADNPersonalStreamCommand

ADNCommand

ADNPersonalStreamCommand

ADNCommand



ADNPersonalStreamCommand

ADNCommand

ADNPersonalStreamCommand

ADNPostStatusCommand

ADNRetrieveChannelCommand

ADNUserLookupCommand

Template

MAD LIBS

Sarah Lake Edition

Greetings, _____ family!
(Last name)

I _____ you a Merry Christmas and a _____ New Year!
(verb) (adjective)

It's time for the yearly wrap-up of _____'s life, which I'm sure
(favorite person)

you all wait for in anticipation. In 2011, I _____ a
(verb ending in -ed)

_____ , which means my parents are _____ in
(degree) (verb ending in -ing)

_____ joy that it only took me _____ years. Don't worry, I only
(adjective) (number)

have two _____ left! In other news, I'm still dating _____,
(units of time) (male name)

living in _____ with _____, the _____
(city) (female name) (superlative)

_____ in the world. It's been way too _____ in
(animal) (adjective)

_____, but it's finally starting to feel like _____.
(US state) (season)

Wishing you a _____ holiday!
(superlative)

In _____,
(emotion)

Sarah (Your _____ _____)
(adjective) (relation to you)

Want to share your funny Mad Libs or see what's *really* up with Sarah's life?
Visit her blog at **http://lovelovelovesar.blogspot.com** and leave a comment!

```objc
//AppDelegate.h
@interface AppDelegate : UIResponder<UIApplicationDelegate>

@property (nonatomic, strong, readwrite) IBOutlet UIWindow *window;
@property (nonatomic, strong, readonly) WebService *webService;

@end
```

```objc
//AppDelegate.m
- (UIResponder *)nextResponder;
{
    return self.webService;
}
```

```objc
@interface WebService : UIResponder

@property (nonatomic, copy, readonly) NSURL *baseURL;
@property (nonatomic, strong, readonly) TokenStorage *tokenStorage;
//..more properties

- (id) initWithBaseURL:(NSURL *)baseURL;
- (void) sendCommand:(WebServiceCommand *)command;

- (void) presentMessageFromCommand:(NSString *)message;
- (BOOL) isAuthenticated;

- (BOOL) performAction:(SEL)action from:(id)sender;
- (BOOL) performAction:(SEL)action from:(id)sender parameters:
(NSDictionary *)parameters;

+ (BOOL) isNetworkReachable;

@end
```

```objc
@interface WebService (Commands)

- (IBAction) loginToService;
- (IBAction) logoutFromService;
- (IBAction) retrieveUsersPersonalStream:(id)sender;

@end
```

```objc
- (void) loginToService
{
    WebServiceLoginCommand *command = [WebServiceLoginCommand
commandWithService:self];
    command.tokenStorage = self.tokenStorage;
    command.username = [self.delegate username];
    command.password = [self.delegate password];
    [command send];
}
```

```objc
- (NSString *)username;
{
    return [self textValueForCellAtIndexPath:[NSIndexPath
indexPathForRow:0 inSection:0]];
}

- (NSString *)password;
{
    return [[self textValueForCellAtIndexPath:[NSIndexPath
indexPathForRow:1 inSection:0]] lowercaseString];
}

- (IBAction) login:(id)sender;
{
    if ([[self username] length] && [[self password] length])
    {
        self.messageLabel.text = @"";
        [[UIApplication sharedApplication]
performActionInResponderChain:@selector(resignFirstResponder)
from:self];
        [SVProgressHUD showWithStatus:@"Logging in..."
maskType:SVProgressHUDMaskTypeBlack];
    }
}
```

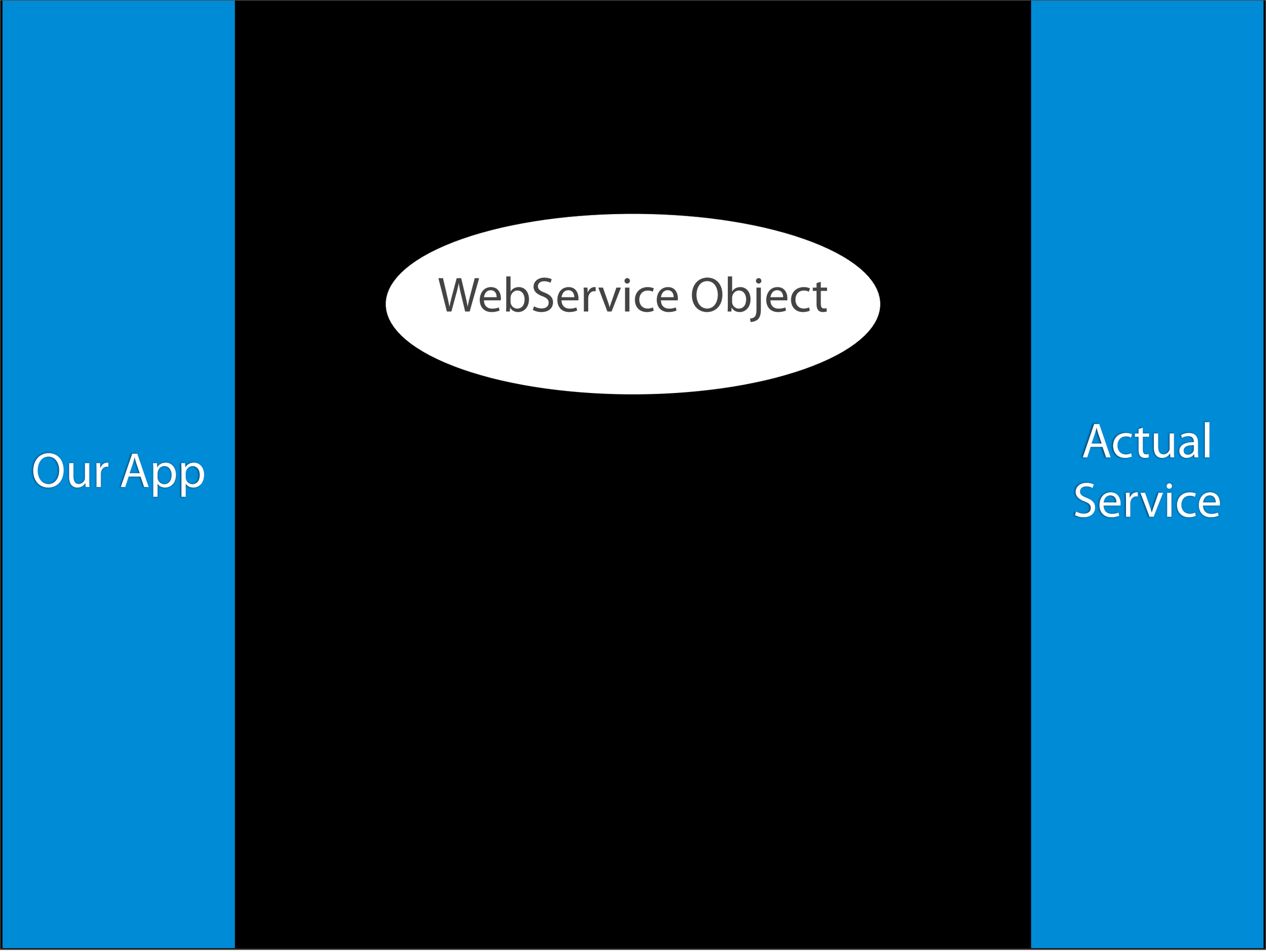
```
- (void) sendCommand:(WebServiceCommand *)command;
{
    BOOL commandVerified = [self verifyCommand:command];
    if (!commandVerified) return;

    if ([self.httpClient networkReachabilityStatus] ==
AFNetworkReachabilityStatusNotReachable)
    {
        [self.delegate displayMessage:@"No Internet Connection"];
        DDLogInfo(@"Not connected to a network");
    }
    else
    {
        AFHTTPRequestOperation *operation = [command createRequestOperation];

        [self.httpClient enqueueHTTPRequestOperation:operation];

        DDLogInfo(@"Sent Command: %@", command);
    }
}
```

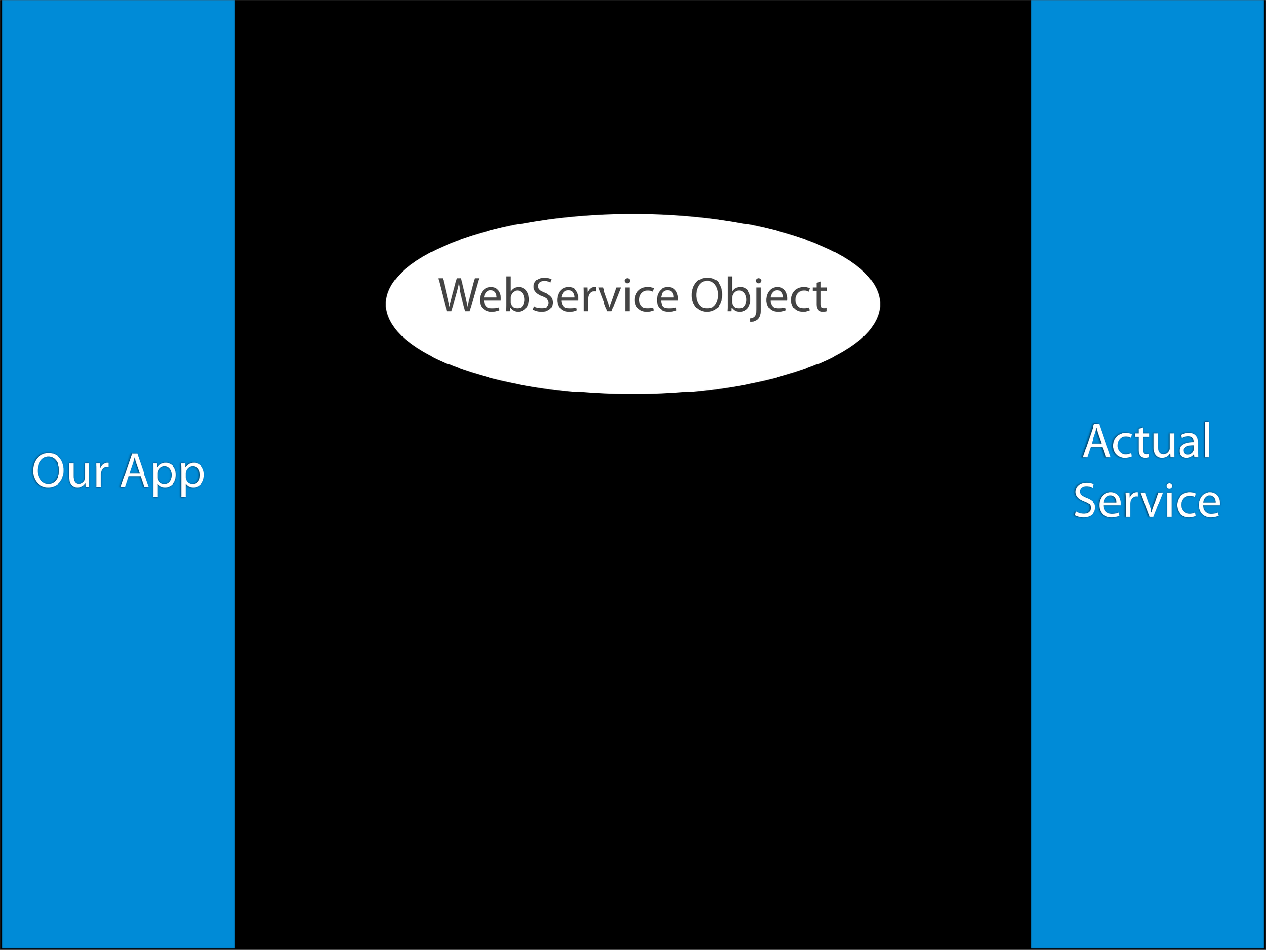
```
- (void) sendCommand:(WebServiceCommand *)command;
{
    BOOL commandVerified = [self verifyCommand:command];
    if (!commandVerified) return;

    if ([self.httpClient networkReachabilityStatus] ==
AFNetworkReachabilityStatusNotReachable)
    {
        [self.delegate displayMessage:@"No Internet Connection"];
        DDLogInfo(@"Not connected to a network");
    }
    else
    {
        AFHTTPRequestOperation *operation = [command createRequestOperation];

        [self.httpClient enqueueHTTPRequestOperation:operation];

        DDLogInfo(@"Sent Command: %@", command);
    }
}
```

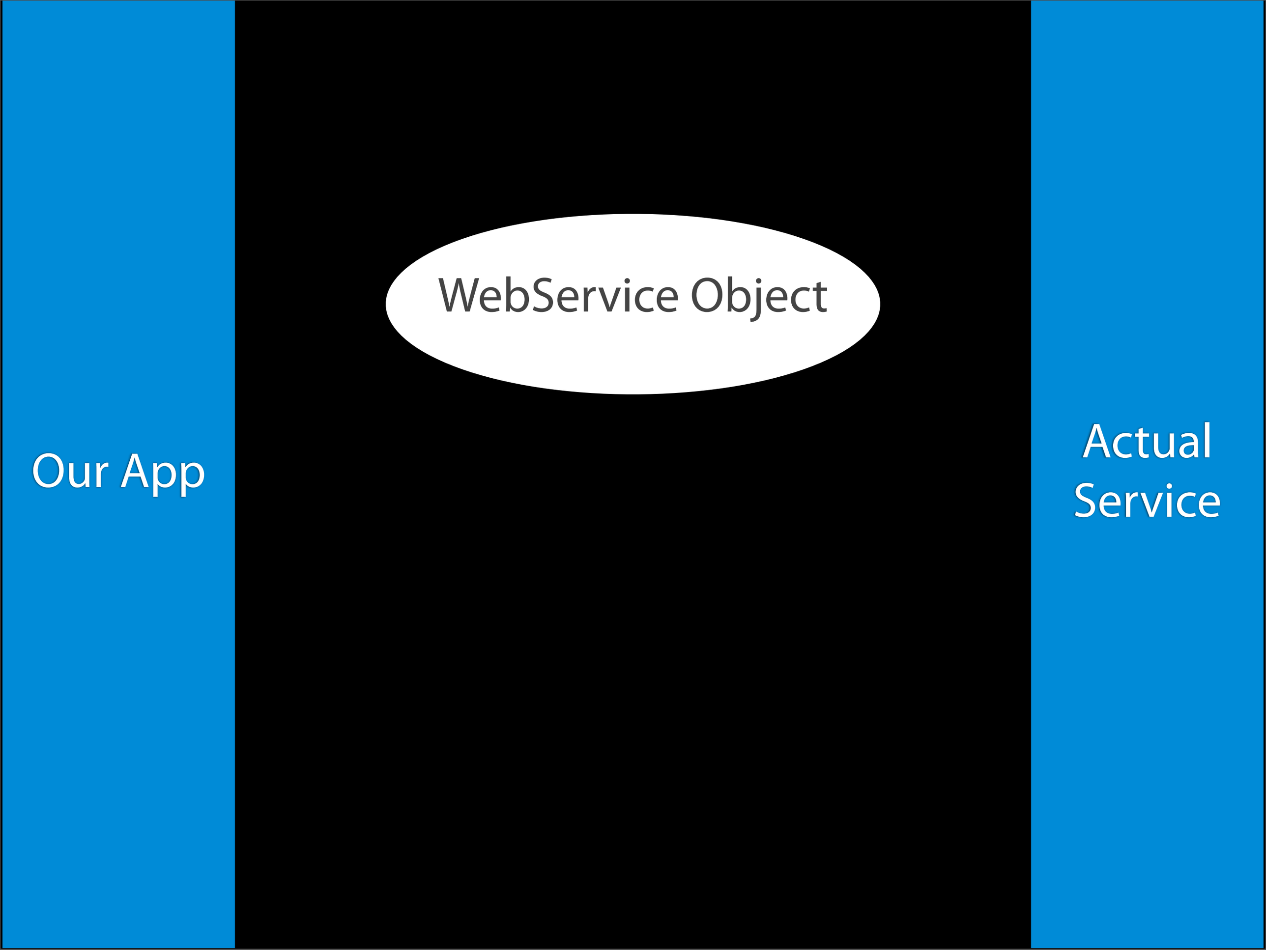
```objc
- (void) sendCommand:(WebServiceCommand *)command;
{
    BOOL commandVerified = [self verifyCommand:command];
    if (!commandVerified) return;

    if ([self.httpClient networkReachabilityStatus] ==
AFNetworkReachabilityStatusNotReachable)
    {
        [self.delegate displayMessage:@"No Internet Connection"];
        DDLogInfo(@"Not connected to a network");
    }
    else
    {
        AFHTTPRequestOperation *operation = [command createRequestOperation];

        [self.httpClient enqueueHTTPRequestOperation:operation];

        DDLogInfo(@"Sent Command: %@", command);
    }
}
```

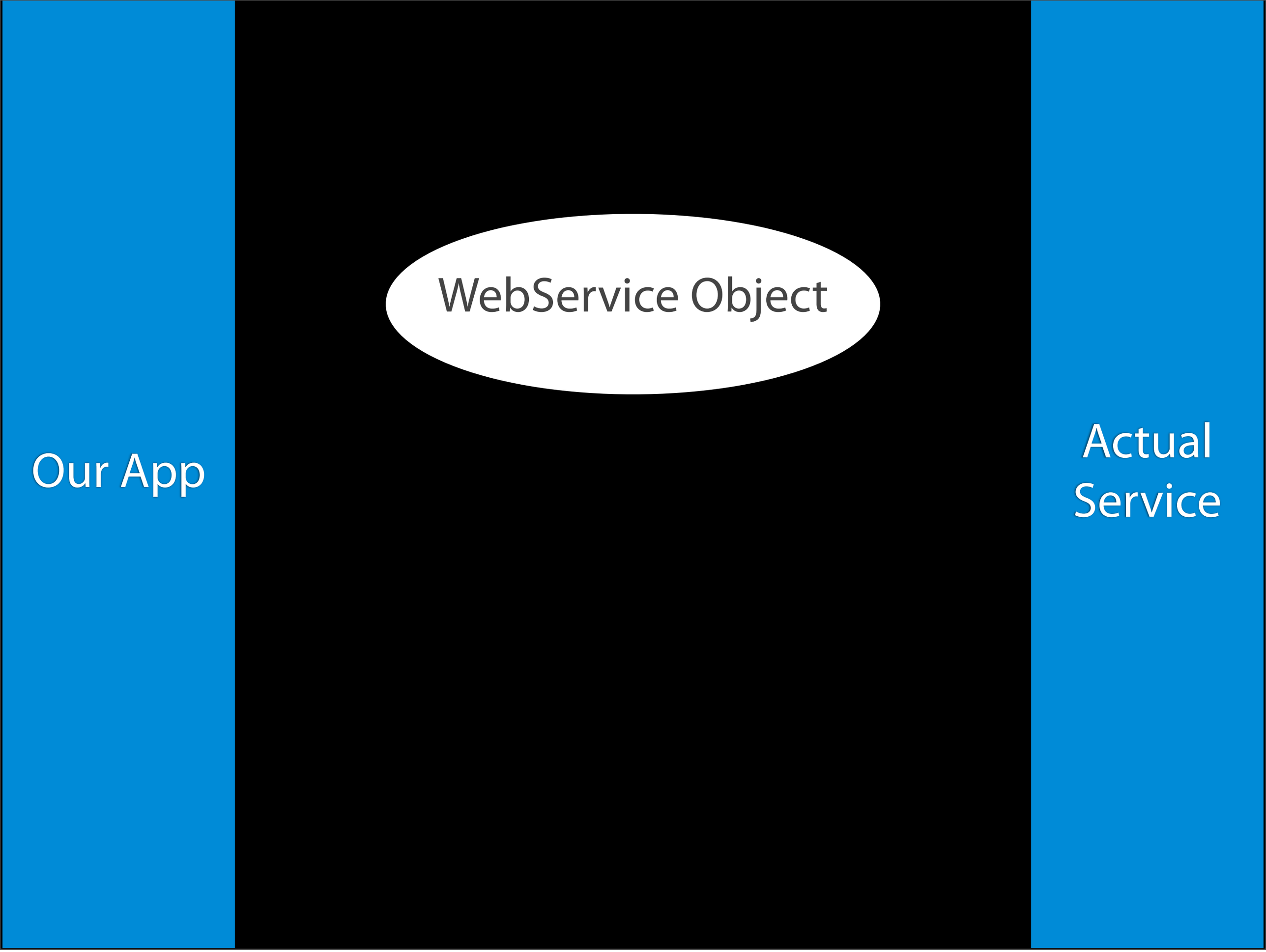
```
[self.delegate displayMessage:@"No Internet Connection"];
DDLogInfo(@"Not connected to a network");
```

```
AFHTTPRequestOperation *operation = [command
createRequestOperation];

[self.httpClient enqueueHTTPRequestOperation:operation];

DDLogInfo(@"Sent Command: %@", command);
```


WebService Object
Our App
Actual Service


WebService Object
Our App
Actual Service


WebService Object
Our App
Actual Service

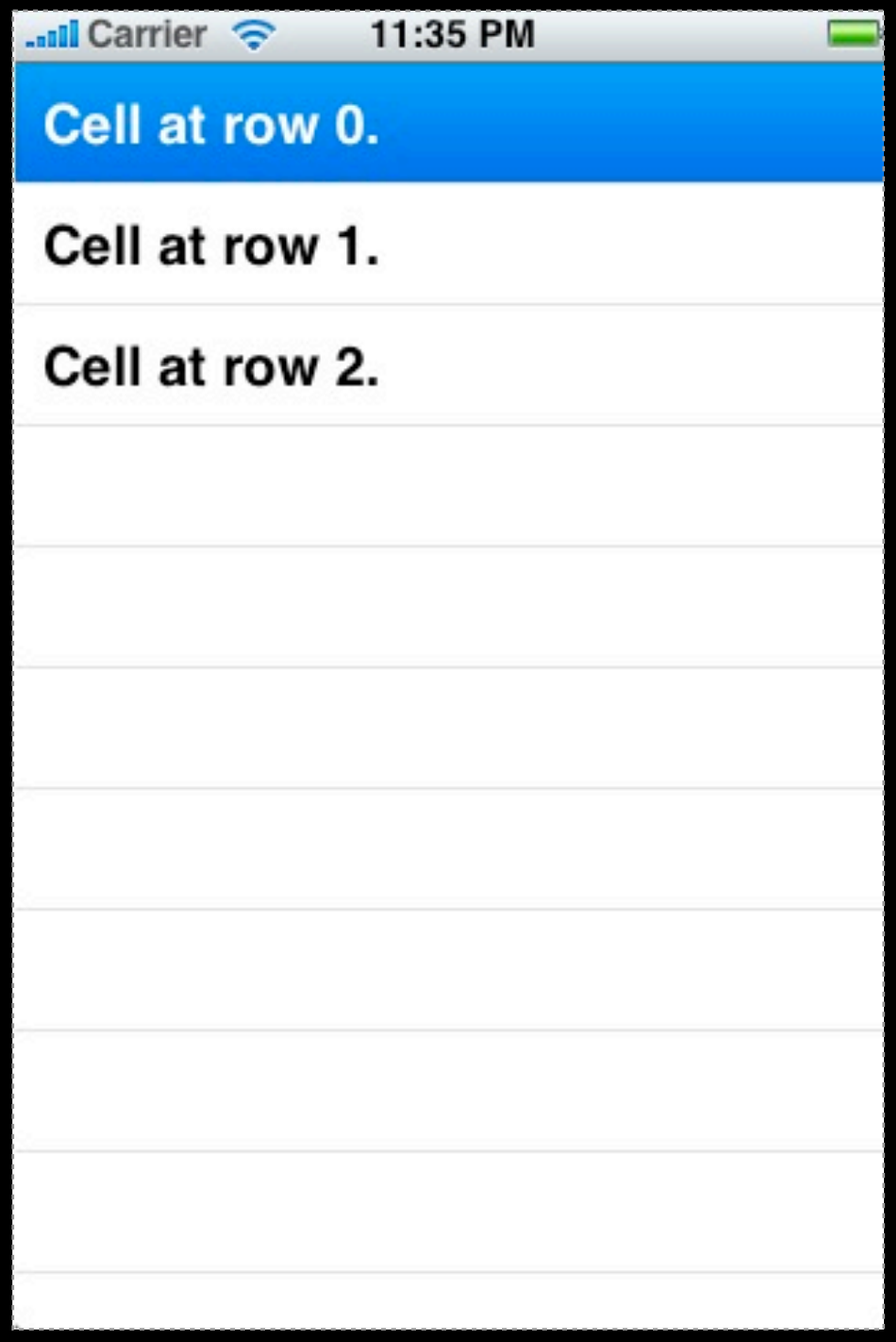

WebService Object
Our App
Actual Service

Needs to be more clear

Usable Object

Usable Object

Needs to be more clear


Carrier
11:35 PM
Cell at row 0.
Cell at row 1.
Cell at row 2.

NSFetchedResultsController

```objc
- (void) loadUserStream;
{
    self.results = [StreamEvent MR_fetchAllSortedBy:StreamEvent.createdDate
                                          ascending:YES
                                      withPredicate:[self streamFilter]
                                            groupBy:nil
                                           delegate:self
                                          inContext:self.context];
}
```

```
- (void) loadUserStream;
{
    self.results = [StreamEvent MR_fetchAllSortedBy:StreamEvent.createdDate
                                           ascending:YES
                                       withPredicate:[self streamFilter]
                                             groupBy:nil
                                            delegate:self
                                           inContext:self.context];
}
```

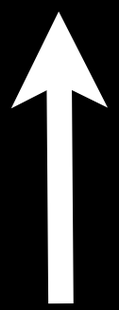
NSFetchedResultsControllerDelegate

Delegation

AbstractTemplate

ConcreteTemplate

AbstractTemplate



ConcreteTemplate

AbstractTemplate

ConcreteTemplate

AbstractTemplate

ConcreteTemplate

```
- (void)controllerWillChangeContent:(NSFetchedResultsController
*)controller



- (void)controller:(NSFetchedResultsController *)controller
   didChangeObject:(id)anObject
       atIndexPath:(NSIndexPath *)theIndexPath
     forChangeType:(NSFetchedResultsChangeType)type
      newIndexPath:(NSIndexPath *)newIndexPath



- (void)controllerDidChangeContent:(NSFetchedResultsController
*)controller
```

```
    switch(type)
    {
        case NSFetchedResultsChangeInsert:
            [tableView insertRowsAtIndexPaths:@[theIndexPath]
withRowAnimation:UITableViewRowAnimationNone];
            break;

        case NSFetchedResultsChangeDelete:
            [self verifyProductPredicate:anObject];
            [tableView deleteRowsAtIndexPaths:@[theIndexPath]
withRowAnimation:UITableViewRowAnimationFade];
            break;

        case NSFetchedResultsChangeUpdate:
        {
            id object = [self.results objectAtIndexPath:theIndexPath];
            [self configureCell:(id)[tableView
cellForRowAtIndexPath:theIndexPath] forObject:object];
        }
            break;

        case NSFetchedResultsChangeMove:
            [tableView deleteRowsAtIndexPaths:@[theIndexPath]
withRowAnimation:UITableViewRowAnimationFade];
            [tableView insertRowsAtIndexPaths:@[theIndexPath]
withRowAnimation:UITableViewRowAnimationFade];
            break;
    }
```

Last steps from controller to view with observer pattern

Observer

View

Controller

Model

Composite

Command

Mediator

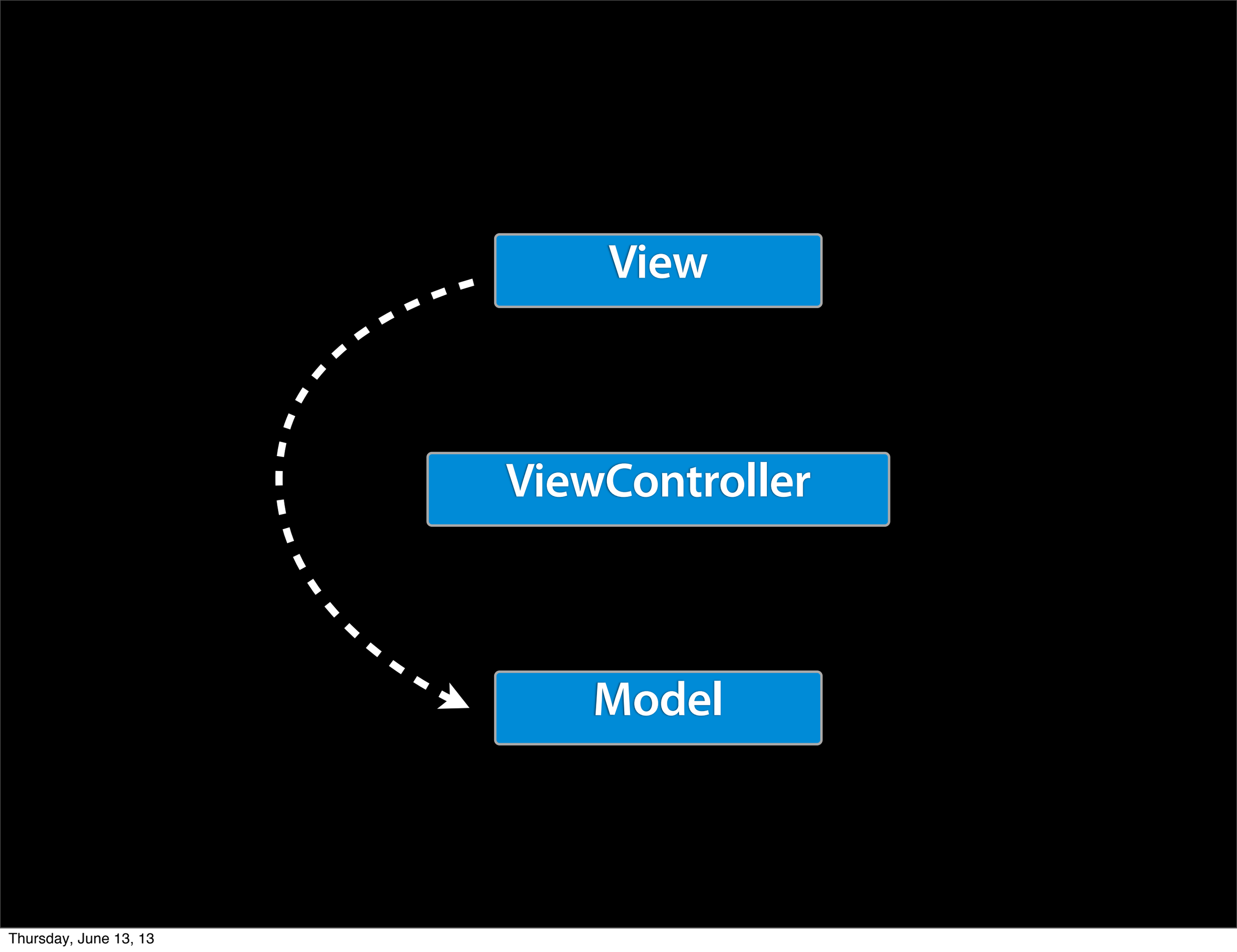
Strategy

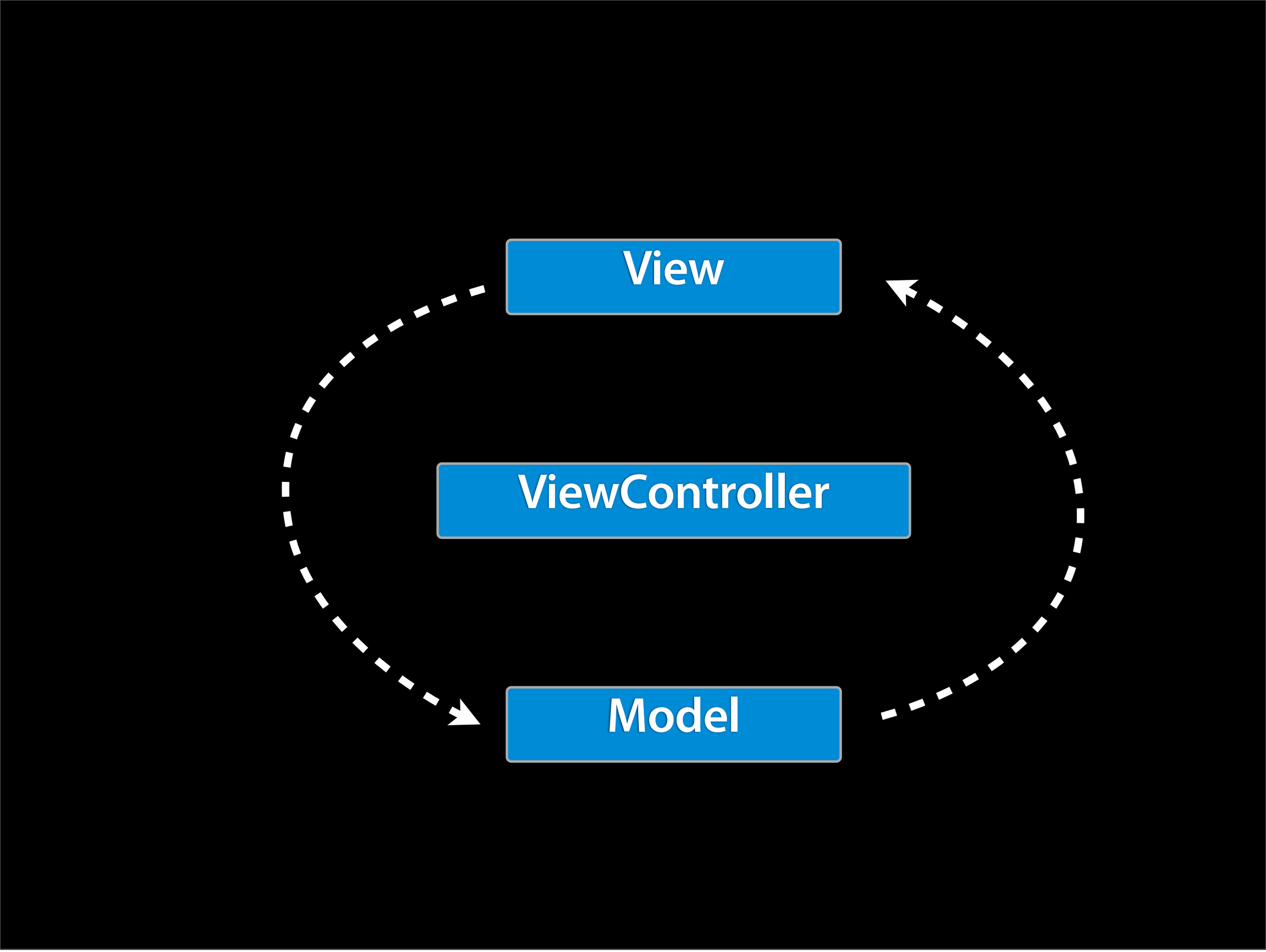
Observer

View

ViewController

Model

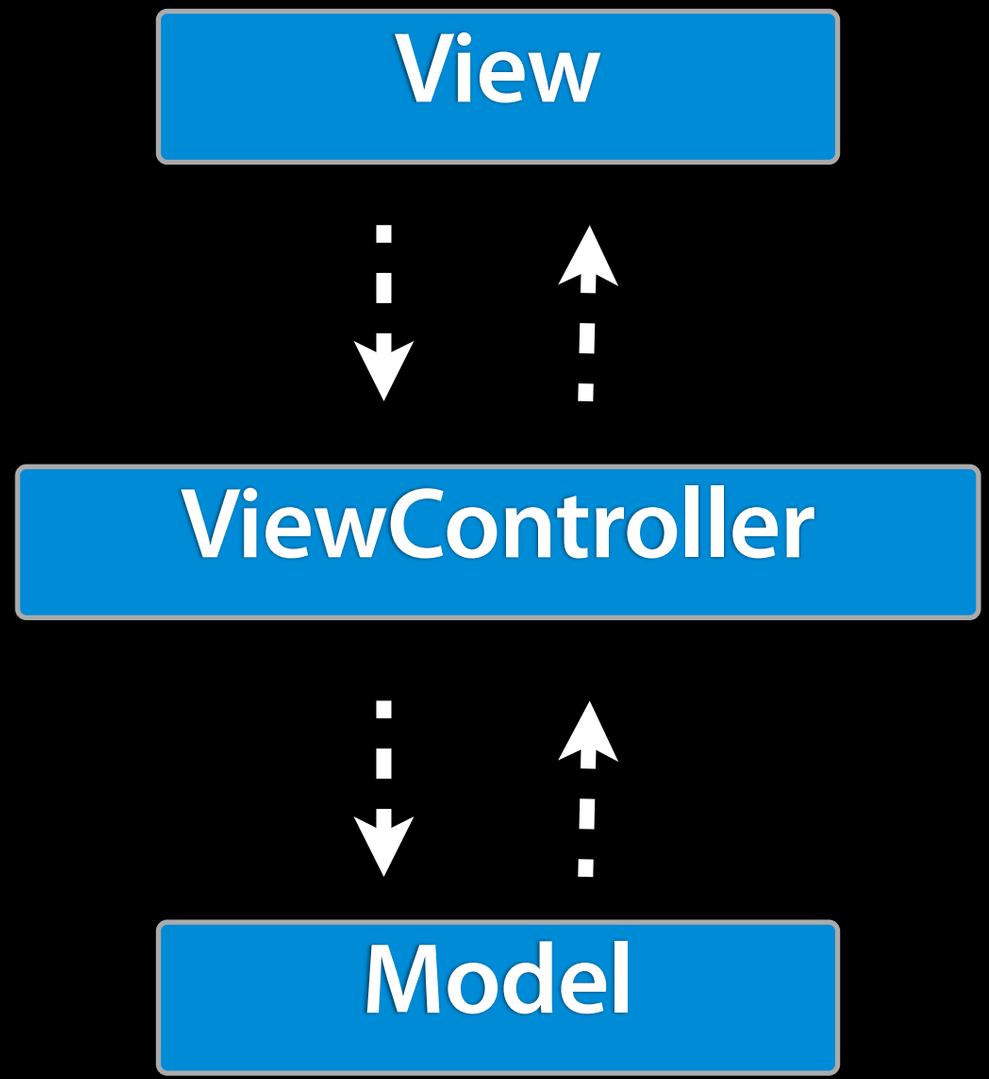

View
ViewController
Model


View
ViewController
Model

View

ViewController

Model


View
ViewController
Model

Touch Interface

View

ViewController

AppDelegate

CoreData

App.net

App.net

Touch Interface

View

ViewController

AppDelegate

CoreData

App.net

App.net

Touch Interface

View

ViewController

AppDelegate

CoreData

App.net

App.net

Touch Interface

View

ViewController

AppDelegate

CoreData

App.net

App.net

Touch Interface

View

ViewController

AppDelegate

CoreData

App.net

App.net

Touch Interface

View

ViewController

AppDelegate

CoreData

App.net

App.net

Touch Interface

AppDelegate

CoreData

App.net

App.net

Touch Interface

View

ViewController

AppDelegate

CoreData

App.net

App.net

Touch Interface

View

ViewController

AppDelegate

CoreData

App.net

App.net

Touch Interface

View

ViewController

AppDelegate

CoreData

App.net

App.net

Touch Interface

View

ViewController

AppDelegate

CoreData

App.net

App.net

Touch Interface

View

ViewController

AppDelegate

CoreData

App.net

App.net

Pattern Abuse

Factory

Abstract Factories

Two-Stage Creation

[[MyClass alloc] init]

NSClassFromString

[NSClassFromString(@”MyClass”) alloc] init]

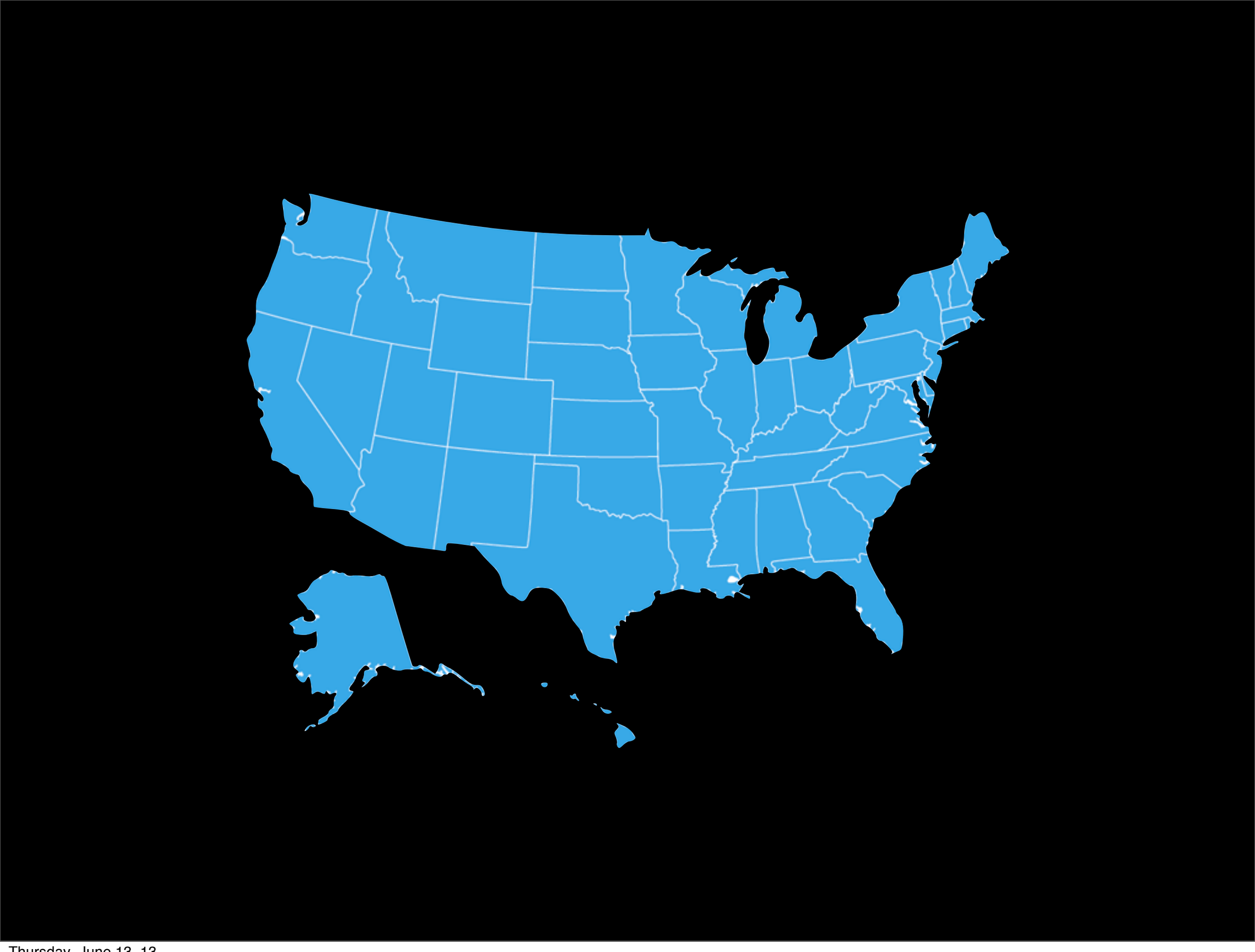
More explanation in slides

Singletons

More explanation in slides

~~Singletons~~

More explanation in slides

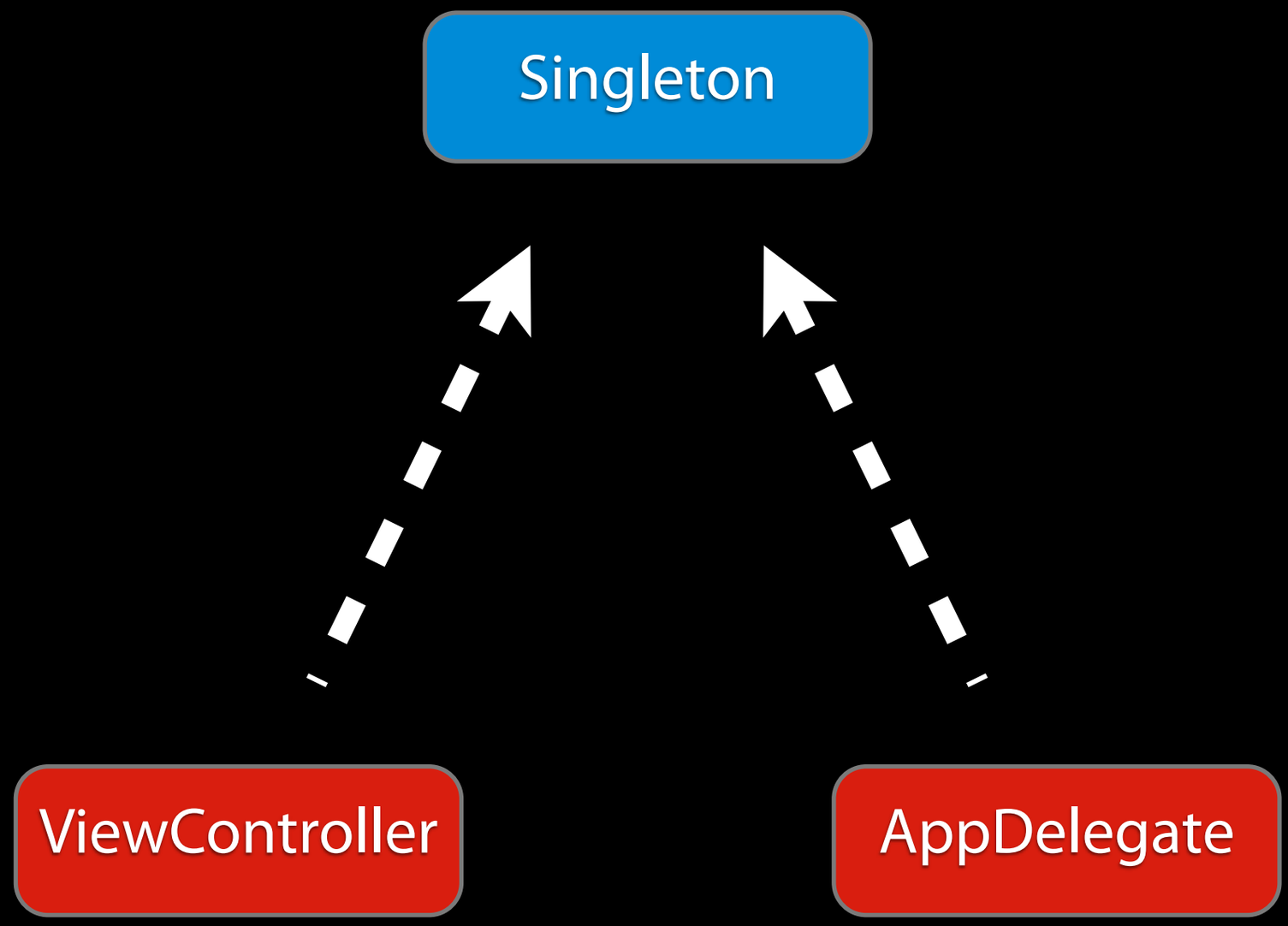


Thursday, June 13, 13

Singleton

ViewController

AppDelegate


Singleton
ViewController
AppDelegate

ViewController

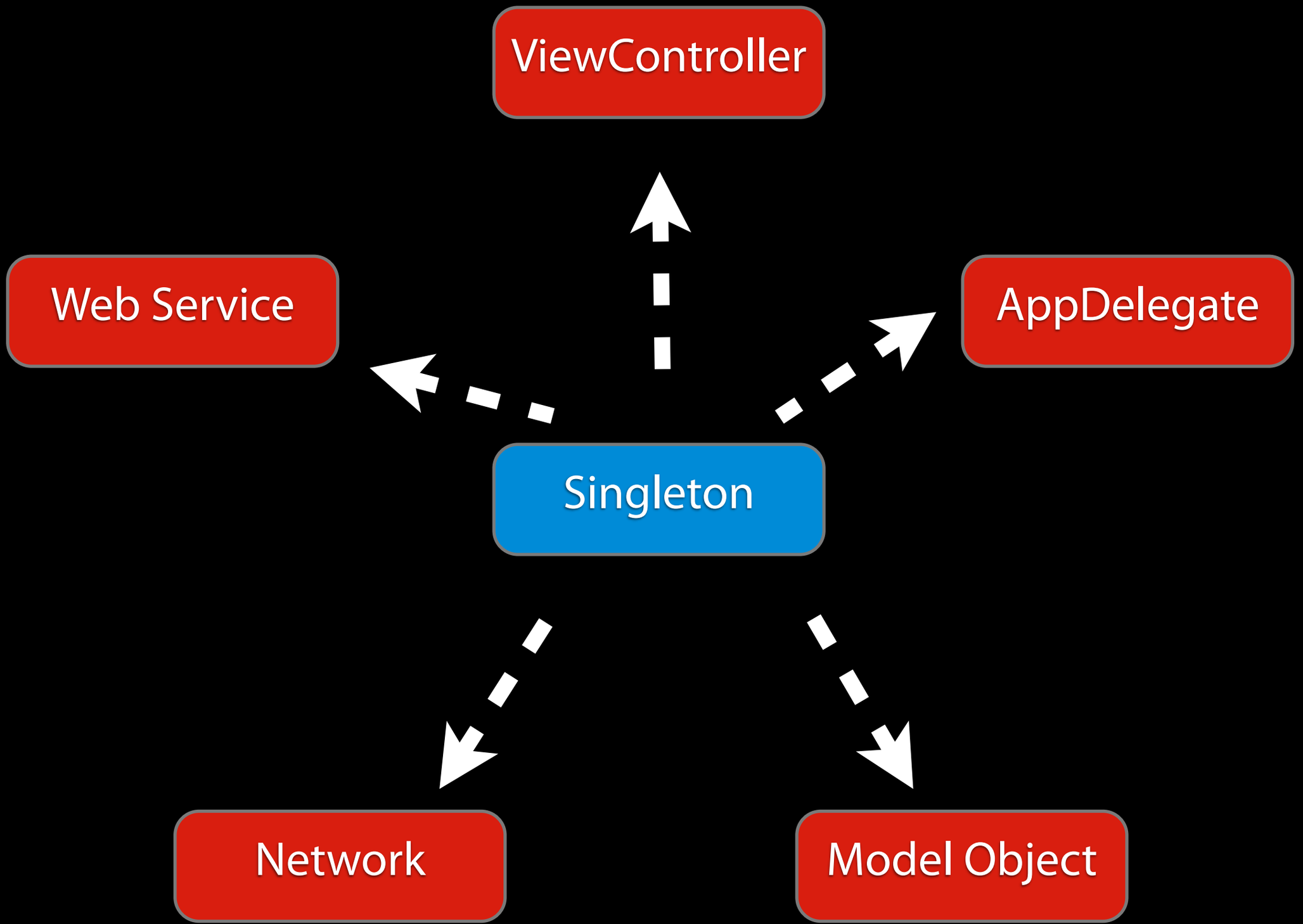
Web Service

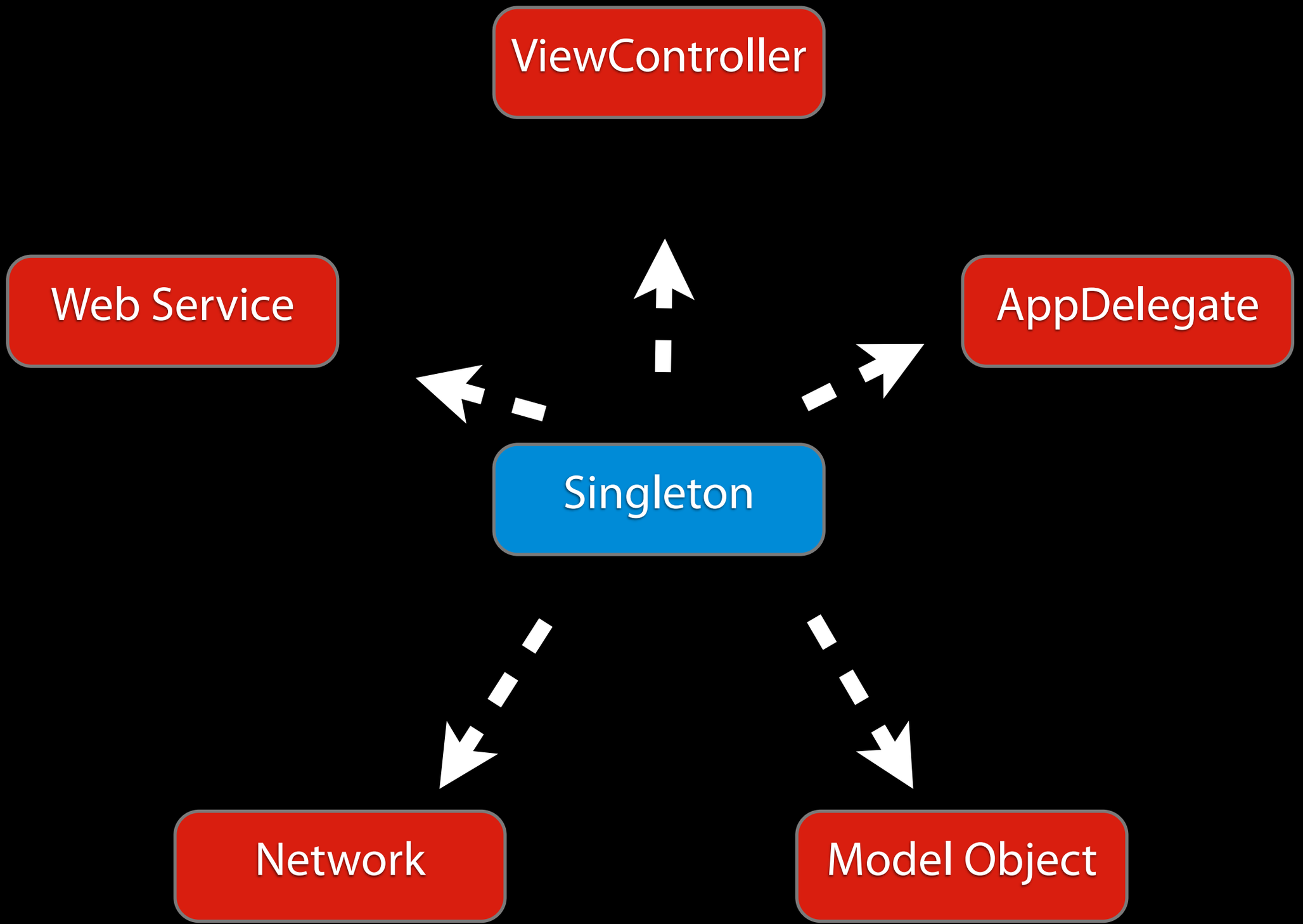
AppDelegate

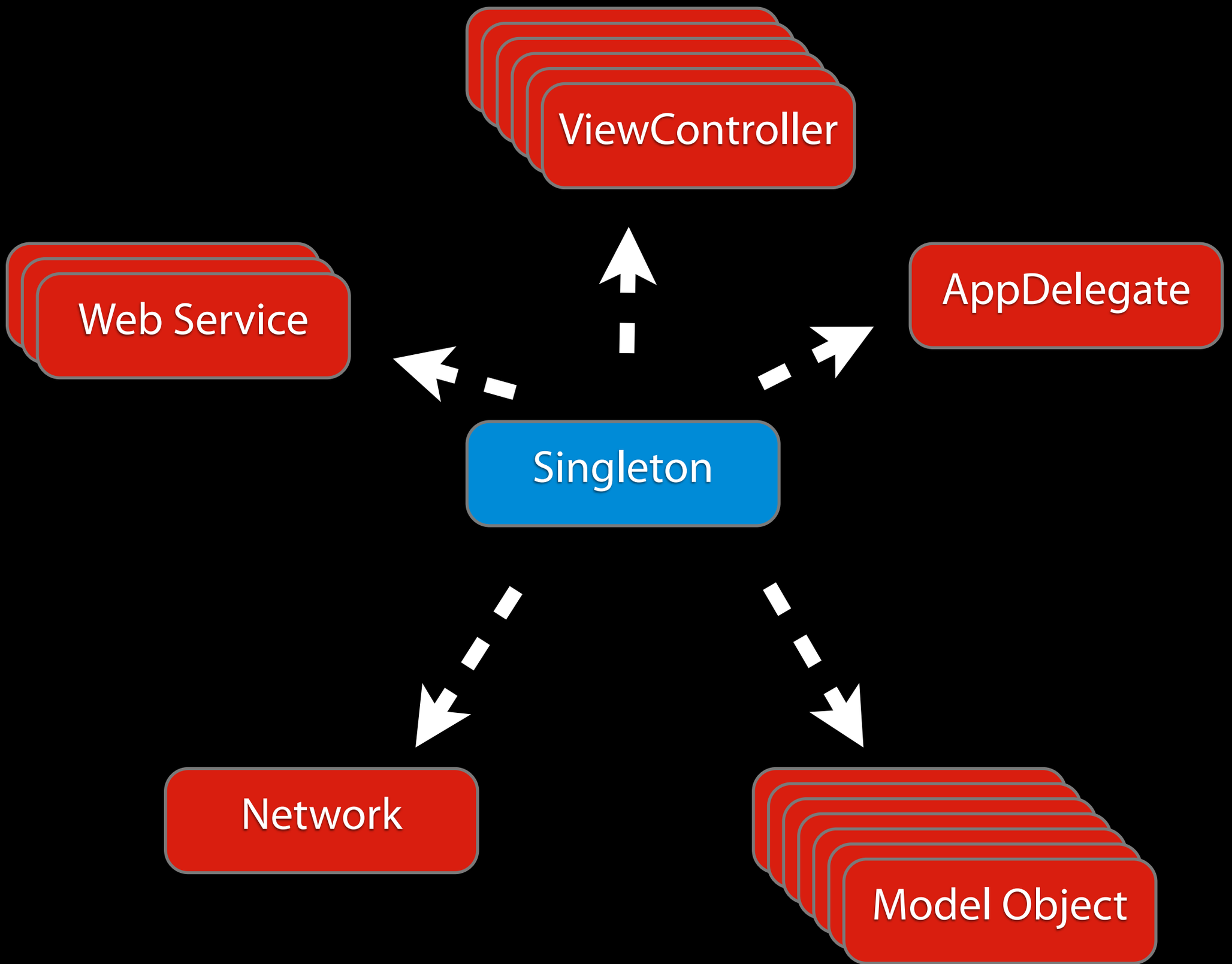
Singleton

Network

Model Object


ViewController
Web Service
AppDelegate
Singleton
Network
Model Object


ViewController
Web Service
AppDelegate
Singleton
Network
Model Object


ViewController
Web Service
AppDelegate
Singleton
Network
Model Object

Singleton

Singleton

AppDelegate

AppDelegate

Singleton Manager

AppDelegate

Singleton Manager

Singleton

Singleton

Singleton

AppDelegate

Singleton Manager

-instanceForClass:[Singleton class]

AppDelegate

Singleton Manager

AppDelegate

Singleton Manager

-destroyInstanceForClass:[Singleton class]

AppDelegate

Singleton Manager

AppDelegate

Singleton Manager

ViewController

AppDelegate

Singleton Manager

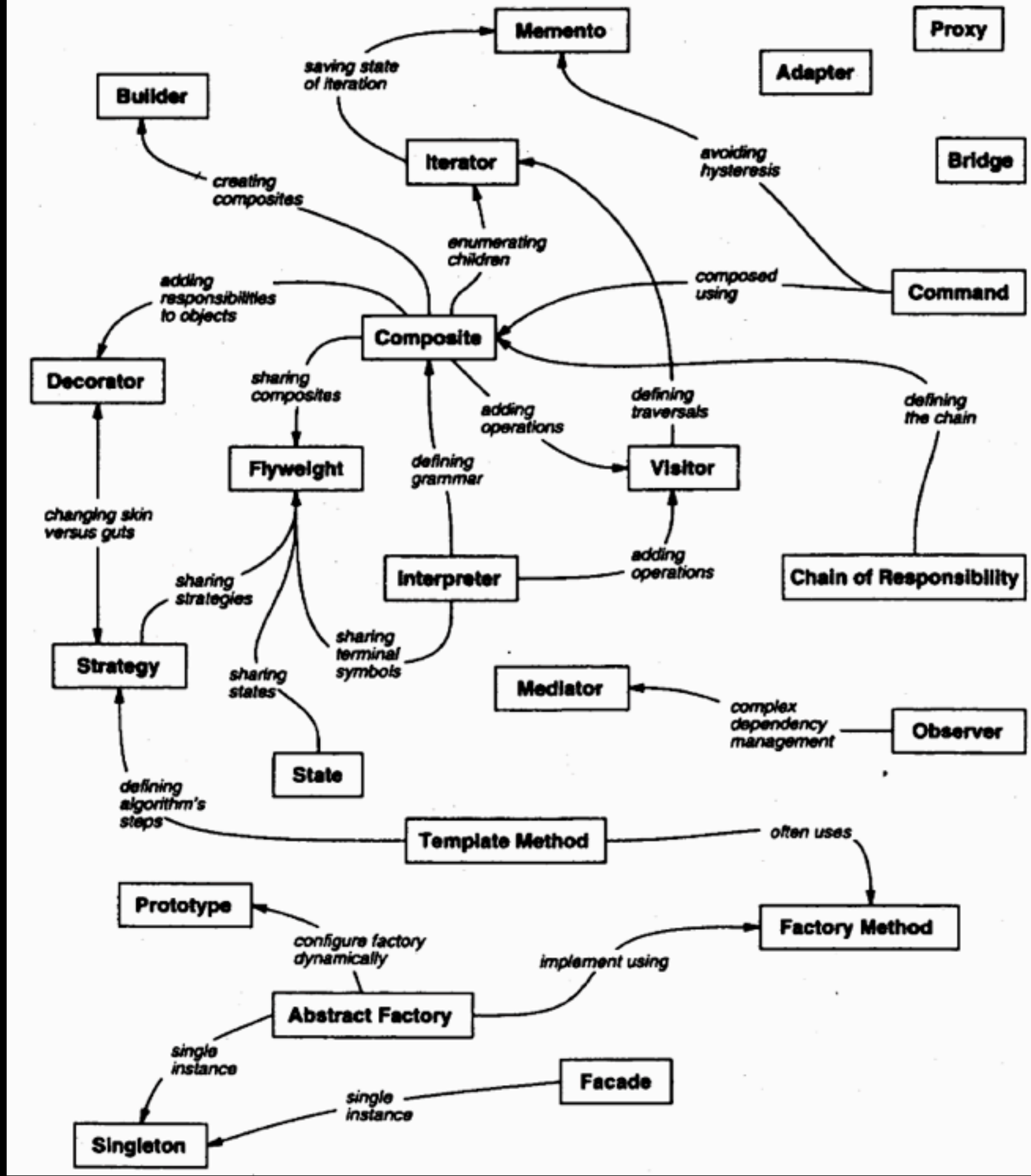
ViewController

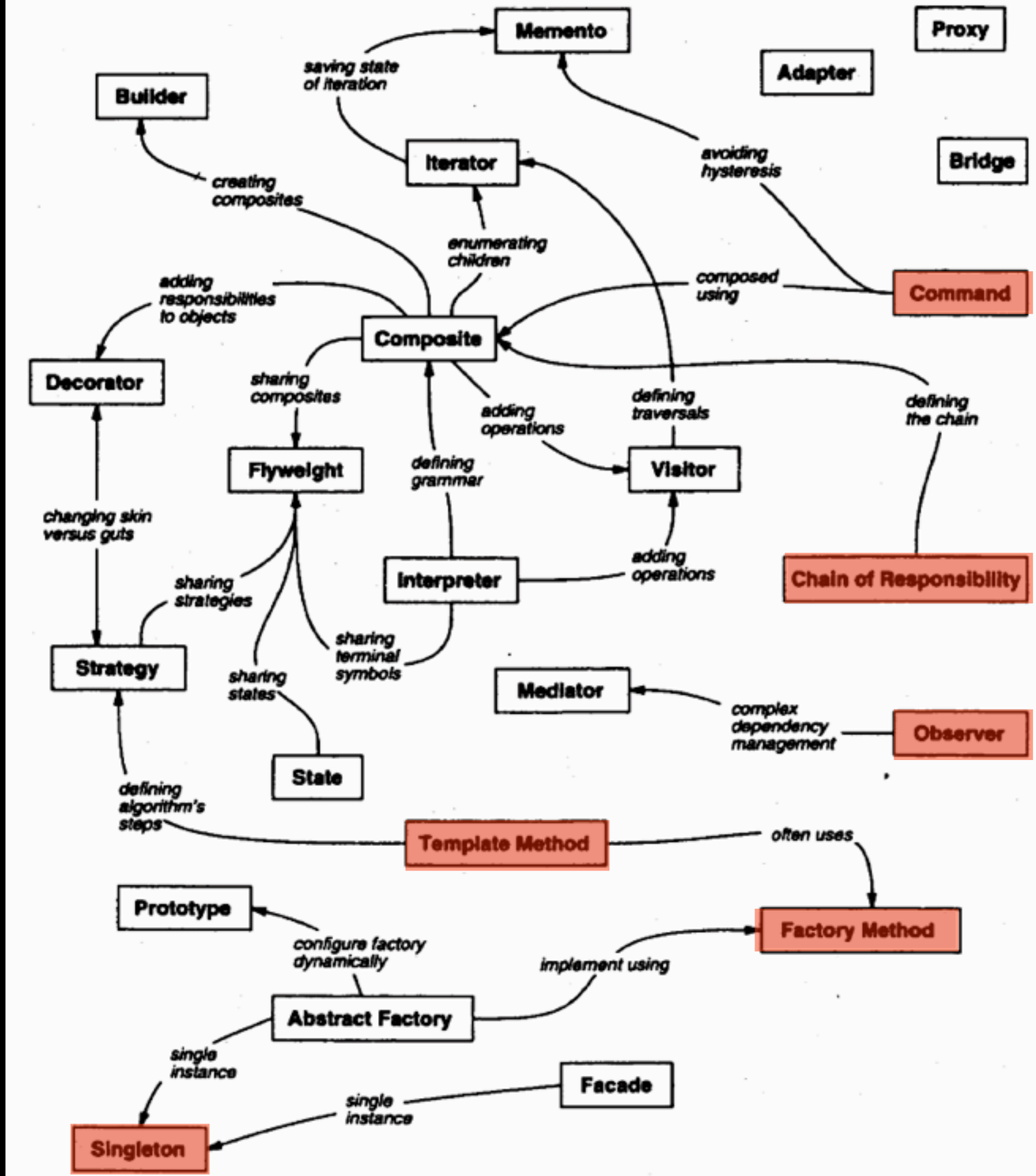
ServiceLocator

Classes are objects too...

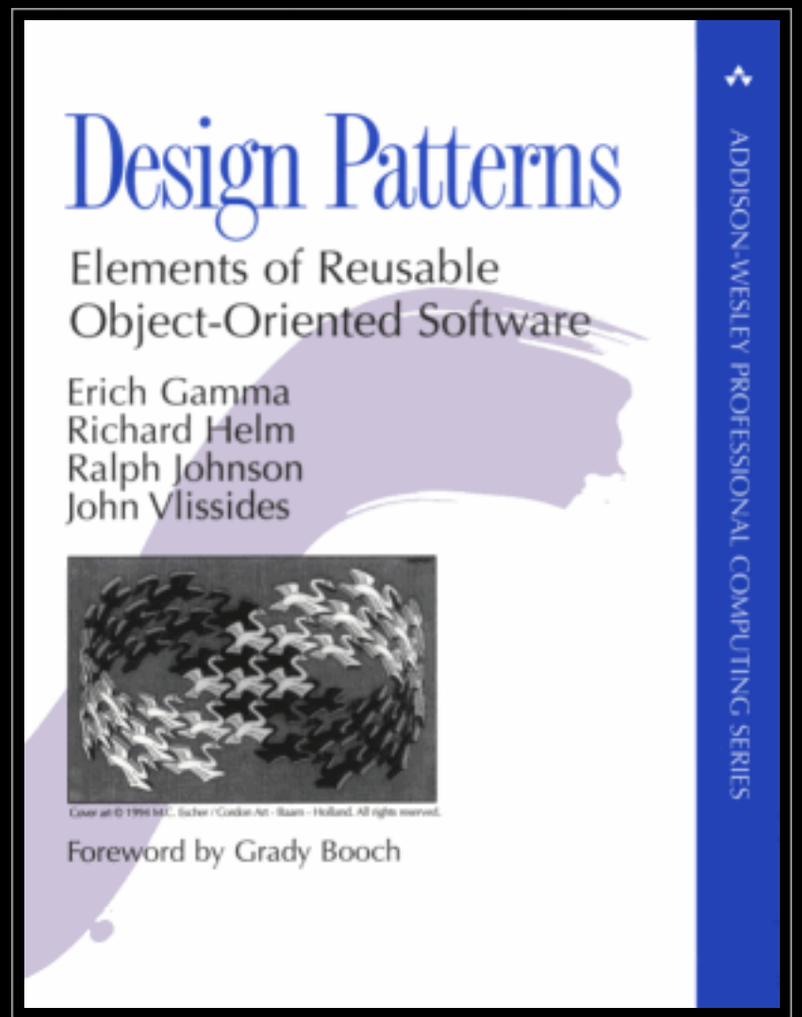
@property Class dateCreator;

```
[self.dateCreator date];
```


Memento
Proxy
saving state of iteration
Adapter
Builder
Iterator
avoiding hysteresis
Bridge
creating composites
enumerating children
adding responsibilities to objects
composed using
Command
Composite
Decorator
sharing composites
adding operations
defining traversals
defining the chain
Flyweight
defining grammar
Visitor
changing skin versus guts
sharing strategies
Interpreter
adding operations
Chain of Responsibility
sharing terminal symbols
Strategy
sharing states
Mediator
complex dependency management
Observer
State
defining algorithm's steps
Template Method
often uses
Prototype
Factory Method
configure factory dynamically
implement using
Abstract Factory
single instance
Facade
single instance
Singleton


Memento
Proxy
Adapter
saving state of iteration
Builder
Iterator
avoiding hysteresis
Bridge
creating composites
enumerating children
adding responsibilities to objects
composed using
Command
Composite
Decorator
sharing composites
adding operations
defining traversals
defining the chain
Flyweight
defining grammar
Visitor
changing skin versus guts
sharing strategies
Interpreter
adding operations
Chain of Responsibility
Strategy
sharing states
sharing terminal symbols
Mediator
complex dependency management
Observer
State
defining algorithm's steps
Template Method
often uses
Prototype
Factory Method
configure factory dynamically
implement using
Abstract Factory
single instance
Facade
single instance
Singleton


SECRET


Design Patterns
Elements of Reusable
Object-Oriented Software
Erich Gamma
Richard Helm
Ralph Johnson
John Vlissides
Foreword by Grady Booch
ADDISON-WESLEY PROFESSIONAL COMPUTING SERIES

Your Brain on Design Patterns

Head First Design Patterns

The Addison-Wesley Signature Series

Patterns of Enterprise Application Architecture

Martin Fowler

The Addison-Wesley Signature Series

Service Design Patterns

Fundamental Design Solutions for SOAP/WSDL and RESTful Web Services

Robert Daigneau

With a Contribution by

Ian Robinson

Foreword by Martin Fowler and Ian Robinson

The Addison-Wesley

Enterprise Integration Patterns

Designing, Building, and Deploying Messaging Solutions

Patterns for Parallel Programming

Timothy G. Mattson
Beverly A. Sanders
Berna L. Massingill

Software Patterns Series

Java Design Patterns

A Tutorial

James W. Cooper

Design Patterns In C#

Cocoa Design Patterns

Erik M. Buck
Donald A. Yacktman

Foreword by Aaron Hillegass, author of Cocoa Programming for Mac OS X

Cocoa Design Patterns

Developer's Library

Reusable Object

Martin Fo

Forewords by Ward Cunningham and Ra

Jacobson
Rumbaugh

Patterns Explained

A New Perspective on Object Oriented Design

Second Edition

Alan Shalloway
James R. Trott

Software Patterns Series

It was important, his father said, to craft the backs of cabinets and fences properly, even though they were hidden.

Steve Jobs by Walter Isaacson

“He loved doing things right. He even cared about the look of the parts you couldn’t see.”

- Steve Jobs

saul@magicalpanda.com